Research Memorandum 76-12

PLANIT UTILITY PROGRAM-- OPERATOR/USER MANUAL

Richard F. Bergfeld, James L. Cilva, Barry Seid
James M. Fletcher and Alan M. Hoff
Litton Systems Incorporated

EDUCATIONAL TECHNOLOGY AND TRAINING SIMULATION TECHNICAL AREA

U. S. Army

Research Institute for the Behavioral and Social Sciences

May 1976

DISPOSITION FORM

For use of this form, see AR 340-15, the proponent agency is TAGCEN.

REFERENCE OR OFFICE SYMBOL	SUBJECT
PERI-TP	Clearance and Transmittal of Reports to DTIC

TO DDC-DAA-1
ATTN: Mr. Schrecengost

FROM ARI Rsch Pub Group
For the Behavioral + Soc Sc

DATE 8 Nov 79
Ms Price/48913

CMT 1

1. The reports listed on Inclosure 1 are approved for public release with unlimited distribution (50 numbered ARI Research Memorandums, 74-1 thru 76-30).

2. These are among the previously unrecorded ARI reports which you identified to us 22 June 1979 as not in your retrieval system. The accompanying box contains at least one copy of each report for your retention and reproduction.

Helen S. Price

1 incl
List of reports, 1974-76

HELEN S. PRICE
Research Publications Group
Army Research Institute

x 4

Educational Technology and
Training Simulation
Contract Number DAHC 19-74-C-0064

9 Research Memo[redacted] 76-12

6 # PLANIT UTILITY PROGRAM--
OPERATOR/USER MANUAL.

10 Richard F. Bergfeld, James L. Cilva, Barry Seid,
James M. Fletcher [redacted] Alan M. Hoff
Litton Systems Incorporated

14 ARI-RM-76-12

12 77

Submitted by:

James D. Baker, Chief
Educational Technology and Training Simulation Technical Area

11 May [redacted]76

Approved by:

Joseph Zeidner, Director
Organizations and Systems Research
Laboratory

J. E. Uhlaner, Technical Director
U.S. Army Research Institute for
the Behavioral and Social Sciences

Research Memorandums are informal reports on technical research problems. Limited distribution is made, primarily to personnel engaged in research for the Army Research Institute.

408 010

ABSTRACT

This document presents general operational information and specific procedural data for the operation and use of the PLANIT Utility Program (PUP). PUP is a collection of highly specialized utility routines developed to support the application and use of the AN/GYK-12 (TACFIRE) computer installation of PLANIT (Programming Language for Interactive Teaching.)

The PLANIT Utility Program (PUP) was developed as a part of a Litton Systems Inc., Data Systems Division (DSD), contract with the U. S. Army Research Institute for the Behavioral and Social Sciences (ARI). This contract (# DAHC19-74-C-0064) was awarded on 11 June 1974 as a part of an overall ARI research project which addresses the application of tactical computers to training. This contract specifically addressed the installation of the PLANIT author/student language on the U. S. Army Artillery Tactical Fire Direction System (TACFIRE) general purpose computer. This computer (AN/GYK-12) is also used in several other Army tactical computer systems.

The successful completion of this contract included the delivery and demonstration of a fully operational PLANIT system on the AN/GYK-12 computer. This project included the development of a translator and translation of PLANIT (version 2.6) from FORTRAN to TACPOL (AN/GYK-12 computer programming language). This task was accomplished under a separate ARI contract to the Northwest Regional Educational Laboratory. The Litton contract included the development of the operating system, machine input/output programs, system start and termination routines, utility support programs, and system integration and support to the installation of PLANIT on the AN/GYK-12 system.

BACKGROUND OF THE PLANIT USER TRAINING SYSTEM

Several explicit user requirements converged to generate the research which resulted in the documents contained in this set of reports. The need for some type of user training subsystem in support of tactical automatic data processing (ADP) system developments was clearly established during the evolutionary phase of the Army Tactical Operations System (TOS) development in Europe.[1] In 1974, after a decade of involvement in the development of tactical ADP systems, the Army Computer Systems Command summarized this experience into six "Lessons Learned."[2] One of these lessons was: A dedicated and trained user is required if tactical ADPS is to succeed.

One approach toward meeting this requirement is to apply techniques derived from modern educational technology and the computer sciences by embedding training subsystem packages within the operating system and then using the system itself to teach the user how to use the system. The approach was delineated in a concept paper,[3] which was subsequently submitted, evaluated and found by key Army Personnel to have merit. As a consequence, a requirement was placed on the Army's Behavior and Systems Research Laboratory (BESRL--the predecessor of what is now the Army Research Institute) by what was then the Assistant Chief of Staff for Force Development (ACSFOR) and the Director of Army Research, Office of the Chief of Research and Development (OCRD),[4,5] to effectuate the research necessary to test the concept.

[1] Baker, J. D. "Human Factors Experimentation Within a Tactical Operations System (TOS) Environment." Proceedings: Office of Naval Research Sponsored Tri-Service Coordination Meeting, London, England, 20-21 February 1968.

[2] Memorandum from Headquarters, U.S. Army Computer Systems Command to Assistant Deputy Commander, CACDA, Ft. Leavenworth, KA; Deputy Commander, MASSTER, Fort Hood, TX; Project Manager, Army Tactical Data Systems, Fort Monmouth, NJ, dtd 30 January 1974, Subject: TSDG Lessons Learned.

[3] Memorandum from U.S. Army Behavior and Systems Research Laboratory to Assistant Chief of Staff for Force Development, dated 28 September 1971, Subject: Proficiency Maintenance Using Computer-Assisted Instruction in an Operational Setting.

[4] Memorandum from Assistant Chief of Staff for Force Development to Chief of Research and Development, dated 10 November 1971; with 18 November 1971 indorsement to Behavior and Systems Research Laboratory, Subject: Request for Research in Application of Tactical Data Systems for Training.

[5] Memorandum from Chief of Research and Development to Assistant Chief of Staff for Force Development, dated 29 Nov 1971, Subject: Request for Research in Application of Tactical Data Systems for Training.

The terms of the requirement actually levied, however, went well beyond the scope of the original concept and called for a simultaneous attack on all facets of the problem associated with testing the feasibility of the approach. In terms of broadened scope, the primary role of these systems is in support of tactical operations. Our original concept paper suggested a potential, select secondary role for these computerized tactical data systems, viz., that of directly supporting the system user by using the system itself, in a stand-alone mode, to teach the user how to use the system. The agencies structuring the research requirements saw a possible tertiary role for these systems. About the time they were structuring their requirements, the Army's Dynamic Training Board identified the maintenance of proficiency of Military Occupation Specialty (MOS) 11B40, the light weapons infantryman, as a glaring unit training problem and suggested that Computer-Assisted Instruction (CAI) as one technique for alleviating the situation.[6] In addition, a subsequent Continental Army Command (CONARC) Task Group report on CAI identified the 11B40 MOS as a top contender for attention in the "non-technical" skills area.[7] Consequently, the scope of the effort was expanded to encompass an examination of a tertiary role, i.e., in support of the system's parent unit by using these computers to meet individual and unit training requirements such as those associated with the 11B40 MOS. Additionally, in response to concern that the implementation of the Modern Volunteer Army concept might produce a need for general education development (GED) upgrading it was determined that an examination should be made of the feasibility of employing extant CAI GED on tactical computers in an operational setting. The assumption was made that accomplishment of these latter requirements would be tantamount to proving the feasibility of the secondary role concept as well. The test, therefore, would be a cost-effective undertaking since it would provide data directed toward answering a number of diverse questions concerned with a common training delivery system, viz., tactical computers.

Irrespective of whether it was the secondary or tertiary role concept being assessed, four major components were required: a test in a credible operational environment; appropriate hardware; functioning software and representative people-ware. The vehicle for this overall assessment was MASSTER[8] Test FM 122, "IBCS: Automated Instruction." The hardware was a "given" viz., the Developmental Tactical Operations

[6] Report of the Board for Dynamic Training, Volume II. 17 December 1971, page 116.

[7] Headquarters, United States Continental Army Command Task Group Report and Computer Assisted Instruction. April 1972.

[8] MASSTER - Modern Army Selected Systems Test, Evaluation, and Review--is the Army's test bed for assessing equipment, concepts and doctrine. This activity is located at Fort Hood, Texas.

System (DEVTOS) which was then located at Fort Hood, Texas (Hoyt, et al[9] provide a description of the hardware). Likewise, the people were a "given"--our student population would be MOS 11B40 personnel drawn from the 2nd Armored Division and 1st Cavalry Division located at Fort Hood. The question of what "software" approach to take (specifically, whether to use an existing student/author language) was key to the success or failure of Test 122. Clearly, the decision made at this juncture would determine whether we would hit the assigned "test window" in time to condųct the test. As a related issue, courseware development would largely depend upon the structure of the student/author language selected, so courseware development could not commence until this decision was made. The decision itself had to be correct and timely--and whatever decision was made would undoubtedly be risky.

To add to the difficulty in reaching a decision, it must be realized that it could not be made unilaterally. Conduct of a test of the complexity of MASSTER Test FM 122 required support from and coordination between a number of different agencies--key among them being mutual cooperation of the organization which had DEVTOS responsibility, the U.S. Army Computer Systems Command (USACSC), and the Army Research Institute (ARI). A Memorandum of Understanding[10] was drawn up between these two organizations and, as the first USACSC task in this joint undertaking, a MASSTER Test 122 CAI Concept Paper[11] was to provide alternative concepts for implementing automated instruction materials on the DEVTOS in support of MASSTER Test 122. Concurrent with this effort, a contract was let by ARI with the System Development Corporation (SDC) to develop the courseware (i.e., the instructional materials which would be presented through CAI). The first task SDC had to accomplish was to provide alternative student/author language alternatives for generating the courseware and to determine which alternative provided the best likelihood of success under the test conditions and time constraints imposed. In essence, the combined results of these analytic studies were expressed as follows: "At this stage, many alternative design concepts can be formulated. However, due to time constraints on the implementation of any concept, the only alternative concept considered feasible...is the use of PLANIT."[12]

[9] Hoyt, W. G., Butler, A. K. and Bennik, F. D. "Application of Tactical Data Systems for Training: DEVTOS Feasibility Determination and Selection of an Instructional Operating System." ARI Technical Paper 267, October 1975.

[10] Memorandum of Understanding Between Commander, U.S. Army Research Institute and Commander, U.S. Army Computer Systems Command, Dated 5 June 1973.

[11] Bunker-Ramo Technical Note "MASSTER Test 122--Computer Assisted Instruction (CAI) Concept Paper," February 1973, prepared for the U.S. Army Computer Systems Command.

[12] Ibid. 11, page 18.

PLANIT (Programming Language for Interactive Teaching) is an instructional system consisting of an author language and supporting computer programs for preparing, editing and presenting any subject matter suitable for individualized CAI presentation to students, as well as recording all relevant response data for immediate utilization and subsequent analyses. PLANIT was developed over an eleven year period under the aegis of the National Science Foundation (NSF) at a total investment cost of approximately $740,000. The main goal of this NSF project was to produce a student/author language which would be fully transportable and guaranteed compatible with a large and diversified class of machines.[13] We at ARI take professional pride in the fact that it was our early and subsequent work with PLANIT which validated this visionary transportability notion of NSF.[14] We also take "economic" pride in the fact that we capitalized upon an already "hefty" U.S. Government investment to solve a problem, rather than slipping into the classic mold of "reinventing the wheel" by starting from scratch and building a separate student/author language tailored to the hardware/software system constraints.

To lower the curtain on MASSTER Test FM 122, the test was successfully conducted and demonstrated that it was feasible to use tactical computers in a stand-alone training mode to satisfy individual and unit training requirements. It was found that automated instruction in a field setting was enthusiastically accepted by the non-commissioned officers (NCO's) examined and, as a training medium, it proved to be more effective than the traditional study-method of training.[15,16,17,18,19]

[13] Frye, C. H. "A Report on PLANIT: One Stage of Completion," Final Report for the National Science Foundation Grant No. EPP73-07319 A04, August 1975.

[14] For a complete account of the experiences of ARI in installing, using and evaluating PLANIT in an Army setting, including all the "warts and blemishes" uncovered during this endeavor, see: Johnson, C. "Implementation of PLANIT at the U.S. Army Research Institute for the Behavioral and Social Sciences," PLANIT Newsletter, July 1975.

[15] Hoyt, W. G. and Baker, J. D. The use of tactical computers to provide weapons and tactics training to combat NCO's: Results of a field test. Proceedings: Sixteenth Annual Conference of the Military Testing Association MTA, U.S. Coast Guard Institute, Oklahoma City, OK. 21-25 October 1974.

[16] Hoyt, W. G., Butler, A. K. and Bennik, F. D. Application of tactical data systems for training: Volume II - CAI/DEVTOS automation studies. ARI Technical Paper 267, October 1975.

[17] Hoyt, W. G., Butler, A. K. and Bennik, F. D. Application of tactical data systems for training: Volume I - Executive Summary. ARI Technical Paper ___, in preparation.

But the results of this test proved more than the preceding. They also indicated that the obvious Army needs mentioned at the outset of this preface, could be met by applying this technology to a real and present problem. It also went beyond the exploratory stage and satisfied a specific Army requirement. The U.S. Army Combat Developments Command (CDC)/Systems Analysis Group (now the U.S. Army Training and Doctrine Command/Combined Arms Combat Developments Activity, or TRADOC/CACDA) had levied the following requirement[20] on ARI:

> The Proposed Material Need for the Tactical Operations System - TOS (Unclassified title, portions of contents classified CONFIDENTIAL) states: "During system non-tactical employment the equipment shall have the capability to permit the training of user personnel without affecting the mission ready capability of the system." While the need exists, no specific data are extant which can be brought to bear on this problem. The requested research will provide data which could impact on all TOS users and result in considerable savings in training costs related to the user's need to maintain proficiency in the use of these systems.

The 122 Test data satisfied the CDC requirement. The Proposed Material Need (MN) for TOS was found to be a viable concept and that MN remains to this day as a bonafide component of the TOS program.

As previously discussed, the results from MASSTER Test FM 122 demonstrated the viability of the embedded training subsystem concept in general and that tactical data systems could be used in a tertiary role, i.e., specifically, that these systems could be used in a stand-alone mode in support of individual and unit softskills training requirements. But conceptually our main goal had always been to embed system specific training packages within the operating system itself and then to use the system to teach the user how to use the system--the earlier noted secondary role for these systems.

[18] Hoyt, W. G., Butler, A. K. and Bennik, F. D. Application of tactical data systems for training: Volume III - Development of courseware and analysis of results for MOS 11B40. ARI Technical Paper ___, in preparation.

[19] Hoyt, E. G., Butler, A. K. and Bennik, F. D. Application of tactical data systems for training: Volume IV - Development of courseware and analysis of results for GED math. ARI Technical Paper ___, in preparation.

[20] Letter, DARB-ARB 19 July 1972, Subject: New Research Requirements for the Human Resources Research and Development Program (RCS CSCRD 70 CRI); letter response from CDCSAG-AG1, same subject as above, dated 1 September 1972.

As a follow-on to Test 122, research was initiated under the aegis of the Product Manager, Computer Training Systems (PM CTS) through HRN 75-158 (and, subsequently, HRN 76-195) which tasked ARI to address the problem of reducing the novice user's difficulties by making tactical data systems (e.g., TOS^2, TACFIRE, TSQ-73, etc.) more "approachable" through applications of the embedded training concept.[21]

Because of its stage of development, the fact that its basic central processing unit would serve as the core for other Army Tactical Data Systems (ARTADS) to follow, and the fact that its operator training problems appeared to be amenable to reduction through the application of automated instructional technology, TACFIRE (the Army's field artillery tactical fire control system) was chosen by the PM CTS as the test vehicle for assessing the embedded training subsystems concept. The initial and specific requirements for the TACFIRE research were delineated in HRN 76-193, "Development and Evaluation of PLANIT Based Computer Embedded Training Packages for TACFIRE" which was prepared by personnel of the U.S. Army Field Artillery School, Fort Sill, OK.

Once again we were faced with the dilemma as to whether the best decision would be to develop a tailor-made student/author language smoothly fitted to the hardware/software constraints of the TACFIRE system, or to build upon our already successfully operating PLANIT system and attempt to install it on TACFIRE. The latter approach had many merits, among them: (1) it was an author language system with which we were familiar, while a customized system would be untested, costly and would require an extensive checkout; (2) a customized authoring system would be limited to a given TACFIRE configuration, whereas PLANIT would be transportable to the family of ARTADS systems, and (3) because of PLANIT's machine independent characteristics, courseware could be prepared on commercial computers and, after content checkout, easily installed on the tactical system, whereas a customized approach would tie-up the actual tactical system during courseware preparation.

The effort to install PLANIT on the AN/GYK-12 computer, the results of which are contained in this set of reports, was independently undertaken as Technology Based - Exploratory Development research and not as Advanced Development activity (i.e., it was not done in direct response to an explicit, stated user need). It serves as a classic example of what Dr. Malcolm R. Currie, Director of Defense Research and Engineering (DDR&E) was describing in the following statement to the Second Session of the 94th Congress: "The objective of the Technology Base is the advancement of technology applicable to future systems and subsystem

[21] Human Resource Need (HRN) 75-158, title: "User Training and Proficiency Maintenance in a Tactical Data Systems Environment," submitted as a research requirement for inclusion in the ARI FY 75 Advanced Development Work Program by the Product Manager, Computerized Training System, Fort Monmouth, NJ. HRN 76-195 was a revalidation of the requirements delineated in 75-158 for inclusion in the FY 76 Work Program.

options. These options (or new ideas) usually involve enhanced military capability, reduced cost, increased performance, better reliability and maintainability, more efficient use of resources or some combination of these attributes." Success in this effort would produce a broadly applicable, cost-effective vehicle for employing embedded training subsystem packages in a variety of military system settings.

It merits comment, however, that while this work was a Technology Based-Exploratory Effort, it had the potential for feeding into the Advanced Development program efforts associated with the user tasks presented in HRN 75-158, "User Training and Proficiency Maintenance in a Tactical Data Systems Environment," if the outcome were successful. Consequently, the PM-CTS was appraised of this effort at the outset and he, in turn, coordinated it with the Program Manager, Army Tactical Data Systems (PM ARTADS). During this coordination some valid points of criticism were raised[22] concerning the PLANIT approach. The PM ARTADS recommended that ARI meet with system developers, users and training agencies as soon as sufficient data were available to determine whether, or not, PLANIT would operate on TACFIRE. At that time a determination would be made concerning implementation implications and to assess if, indeed, this were the most effective approach to take, given the potential for impact on TACFIRE system development efforts. In keeping with this recommendation, a Workshop was convened at ARI in Arlington, VA on 1 October 1974 and these items were covered in detail with personnel from all of the suggested groups in attendance. The interaction was found to be most beneficial to all concerned and the concensus of the group was to install the system described in this set of reports on the TACFIRE system at Fort Sill, OK, and to use it as the test vehicle for assessing the embedded training concept on that ARTADS system.

This historic overview of the events leading up to the production of the set of quite specialized reports may seem untoward in view of the projected, limited set of users of these documents. It is, however, a quite meaningful forum for discussing these events. Too frequently the question is raised as to how did a particular research product originate and was it utilized. The intent here is to show that the warp and woof of concepts and coordination, requirements and research are so intertwined that a simple one-to-one relationship (one response, one use) does not tell the story--only a view of the whole cloth will put it into proper perspective. Additionally, it exemplifies a point made in the previously cited presentation by the Director of Defense Research and Engineering to the 94th Congress when he said: "To deploy systems DOD must not only pursue advanced technology but must endure the long years of research required to bring an idea through growth problems to a finished, proven and useful end product."

[22] Memorandum from Product Manager, Computer Training Systems (PM-CTS) to Program Manager, Army Tactical Data Systems (PM-ARTADS) 28 Jan 74, Subject: HRN 75-158 and 1st indorsement from PM-ARTADS to PM-CTS, same subject as above dated 7 February 74.

This set of reports provides detailed instructions for implementation and operation of PLANIT and auxiliary programs on the AN/GYK-12 computer. The set consists of a report on:

- TRANSL - The PLANIT Translator Program: Installation and Application
- PLANIT Support Programs - Operator/user manual
- PLANIT Utility Program - Operator/user manual
- PLANIT Support and Utility Programs -- Test Procedure
- PLANIT Support and Utility Programs - Flow Charts.

The first report contains the information for installing and operating a program which is designed to translate the FORTRAN from the PLANIT system of programs into the TACPOL language for compilation on the AN/GYK-12 computer. The second covers the general and specific aspects of leading and operating PLANIT on the AN/GYK-12 computer. The third document covers the general and specific aspects of operating the PLANIT utility programs which are a specialized group of routines developed to accomplish various tasks in support of the AN/GYK-12 computer installation of PLANIT. The fourth report covers the procedures used to verify that PLANIT Support and Utility Programs are functioning as per specifications. The fifth document provides the detailed flow charts of the computer logic of the PLANIT Support and Utility Programs.

The effort detailed in the first report (i.e., TRANSL) was accomplished under ARI Contract DAHC19-74-C-0038 by the Northwest Regional Educational Laboratory, Portland, Oregon. The other four reports in the series were prepared by the Data Systems Division, Litton Systems Inc., Van Nuys, CA under ARI Contract No. DAHC19-74-C-0064.

TABLE OF CONTENTS

SECTION	TITLE		PAGE
1	INTRODUCTION		1-1
	1.1	Scope	1-1
	1.2	Manual Organization	1-1
	1.3	General Information	1-1
	1.4	User Information-Data Preparation	1-2
	1.5	Operating Procedure	1-2
2	GENERATION OF PLANIT COMMERCIAL OR MLU LOAD TAPES		2-1
	2.1	General Information	2-1
	2.2	User Information-Data Preparation	2-1
	2.3	Operating Procedure	2-2
3	GENERATION OF BOOTABLE MEDIA (DECKS,TAPE) FROM COMPILER OBJECT OUTPUT		3-1
	3.1	General Information	3-1
	3.2	User Information-Data Preparation	3-1
	3.3	Operating Procedure	3-2
4.	PROCESS SPECIAL PLANIT DATA		4-1
	4.1	General Information	4-1
	4.2	User Information-Data Preparation	4-1
	4.2.1	INPUT DATA Processing	4-1
	4.2.2	OUTPUT DATA Processing	4-2
	4.3	Operating Procedure	4-2
5.	UPDATING THE PLANIT OBJECT LIBRARY TAPE		5-1
	5.1	General Information	5-1
	5.2	User Information-Data Preparation	5-1
	5.3	Library Tape Trouble-Shooting Techniques	5-4
	5.4	Operating Procedure	5-7

TABLE OF CONTENTS (Continued)

SECTION		TITLE	PAGE
6	SUMMARIZING THE PLANIT OBJECT LIBRARY TAPE		6-1
	6.1	General Information	6-1
	6.2	User Information-Data Preparation	6-1
	6.3	Operating Procedure	6-1
	6.4	Summarize Output Explanation	6-2
7	PUNCH PLANIT TRANSLATION TAPE		7-1
	7.1	General Information	7-1
	7.2	User Information-Data Preparation	7-1
	7.3	Operating Procedure	7-1
8	CREATE LIBRARIAN INPUT TAPE FROM PLANIT TRANSLATION SOURCE TAPE		8-1
	8.1	General Information	8-1
	8.2	User Information-Data Preparation	8-1
	8.3	Operating Procedure	8-1
9.	GENERAL CHARACTER CONVERSION ROUTINE		9-1
	9.1	General Information	9-1
	9.2	User Information-Data Preparation	9-1
	9.3	Operating Procedure	9-2
	9.4	AN/GYK-12 PLANIT System Character Sets	9-3
APPENDIX A		PUP CONTROL CARDS	A-1
APPENDIX B		SAMPLE SUMMARIZE OUTPUT	B-1
APPENDIX C		COPYING TAPES	C-1
APPENDIX D		SAMPLE JOBSTREAM CARDS	D-1
	D.1	PLANIT Load Tape Generation	D-1
	D.2	Object Library "Cards File" Update	D-5
APPENDIX E		GLOSSARY OF TERMS	E-1

LIST OF ILLUSTRATIONS

FIGURE	TITLE	PAGE
1-1	Jobstream from Tape	1-3
1-2	Mixed Jobstream from Tape and Card Reader	1-3
2-1	Sample Patch Card	2-4
2-2	Sample Program Deck Set -up	2-4
5-1	Sample Update Library Job Deck	5-2
8-1	Volume Organization - Librarian Input Tape	8-2
9-1	Typical Job Deck for PLANIT Lesson Character Conversion	9-4

LIST OF TABLES

TABLE	TITLE	PAGE
2-I	PUP Generation DIG Values	2-5
5-I	Sample Extra Blank Fault	5-6
9-I	PLANIT Character Set Differences - AN/GYK-12 and ARI Commercial Systems	9-5
A-I	PUP Control Cards	A-1
B-I	Sample Summarize Output	B-2
D-I	PLANIT Load Tape Generation Deck	D-2
D-II	Object Library "Cards File" Update Generation Deck	D-6

SECTION 1
INTRODUCTION

1.1 Scope

This manual covers various aspects of using and operating the PLANIT Utility Program (PUP). PUP is a collection of highly specialized utility routines, developed to accomplish particular tasks in support of the AN/GYK-12 computer installation of PLANIT.

1.2 Manual Organization

This manual is organized by devoting a section to each major application. Each section, other than the first, is divided into at least three parts; general information, user information, including data preparation, and PUP operating procedures pertinent to that application. This manual covers the applications, by section as follows:

a. Section 1: General information, jobstream control, loading and execution procedures.

b. Section 2. Generation of PLANIT commercial or MLU load tapes.

c. Section 3. Generation of Bootable media (decks, tape) from Compiler Object Output.

d. Section 4. Process special PLANIT data.

e. Section 5. Updating of the PLANIT Object Library Tape.

f. Section 6. Summarizing the PLANIT Object Library Tape.

g. Section 7. Punch PLANIT Source Tape.

h. Section 8. Create Librarian Input Tape from PLANIT Source Tape.

i. Section 9. General Character Conversion Routine.

1.3 General Information

PUP is designed to accept a user prepared job stream of activities from either the card reader or from commercial tape or both. Data input may also be from cards or tape. A jobstream which is large or which may be run frequently is typically card-to-taped with an off-line program and executed via tape input. Appendix D illustrates some typical jobstreams.

1.4 User Information - Data Preparation

After the user has prepared the required control cards and data for a particular application, the next step is to prepare the jobstream control cards to execute that job. This section of the manual covers only the control cards which directly control the jobstream. Control cards for PUP and their format are given in Appendix A.

The following control cards control the job stream:

a. //TX - This causes commercial tape unit X to be assigned as the input device. X has the value 0, 1, 2 or 3 and is the unit which contains the jobstream to be read.

b. //R - This causes the card reader to be assigned as the input device. If the jobstream is on commercial tape, control will revert to the card reader. If it is encountered as part of a card reader input, the card has no effect.

c. // - This card marks the end of the jobstream and causes tapes to be rewound. The computer will halt (DIG = 770000). The card is located at the end of the jobstream deck.

If the user has determined that the jobstream should be on tape then the regular jobstream input is card-to-taped, in addition a //TX card is prepared to start reading from tape X, see Figure 1-1. Another possibility is a jobstream on tape which picks up data or a job from the card reader, see Figure 1-2 for an example.

1.5 Operating Procedure

To execute a job with PUP, the hardware must be prepared, the program loaded and execution started. The following steps describe this procedure.

a. Hardware setup and loading of PUP

Step	Action
1.	Set the IOU DATA EXCHANGE CHANNEL SELECT switches as follows:

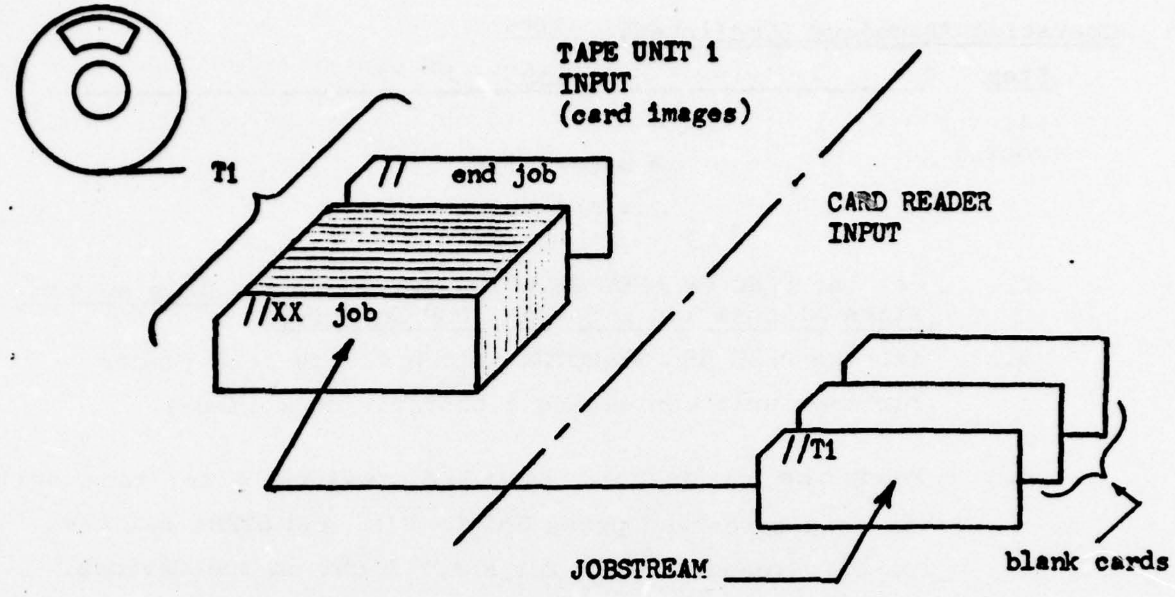


Figure 1-1 . Jobstream from Tape

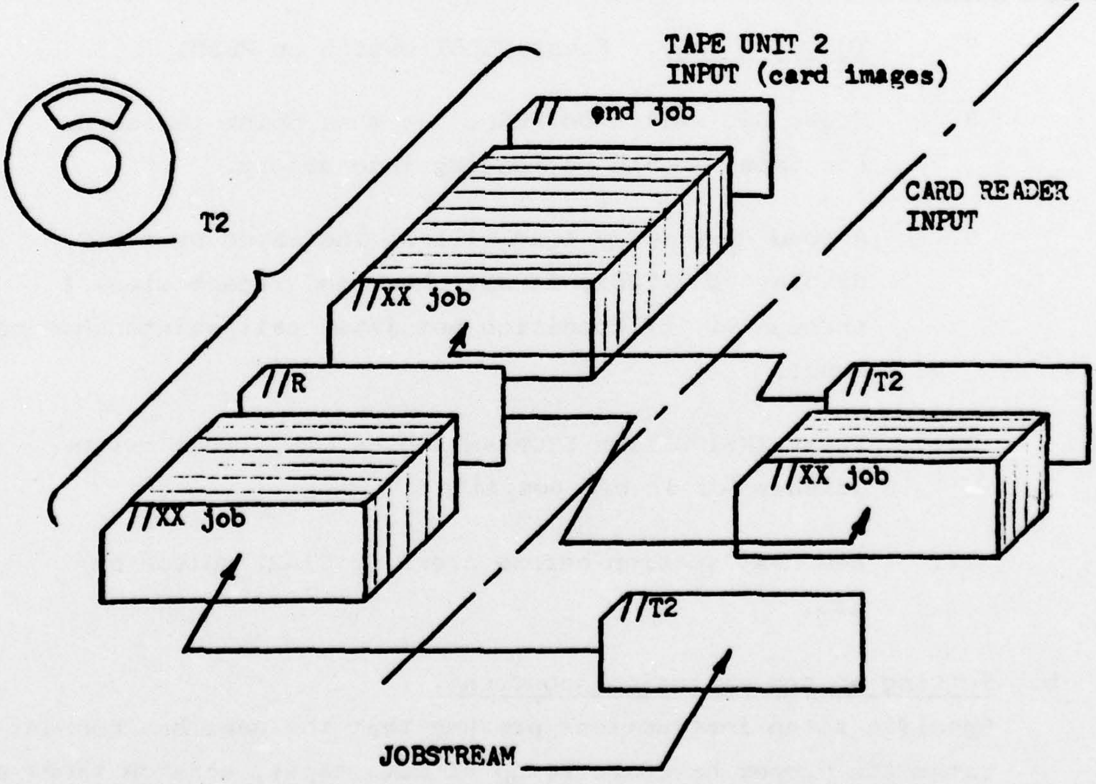


Figure 1-2. Mixed Jobstream from Tape and Card Reader.

1.5 Operating Procedure (Continued)

Step — **Action**

1. (cont.)

A = 2

B = OFF LINE

C = OFF LINE

2. Set the PEBU CHANNEL SELECTOR switch to 1. This switch stays at position 1 for all PUP activity.

3. Set the PEBU BSL SELECTOR switch to the card reader (or tape unit containing a bootable deck image).

4. Ready the BSL device. Load and press READY for tape units or load cards and press END OF FILE and START switches on card reader. Wait for READY light on the devices.

5. Press INSTRUCTION STOP switch on CTS, must be illuminated (or in the on position).

6. Press MASTER RESET switch on CTS.

7. TOS^{2} SSS only. Press RESET switch on PEBU.

8. Press BSL switch on PEBU. At this point the cards (or tape) should be reading into memory.

9. A good bootstrap load will be indicated by a DIG display of 777201, if not check and repeat steps 1 through 9. If condition persists, call maintenance personel.

10. Press INSTRUCTION STOP switch on CTS, light extinguishes (or in off position).

11. See next section before pressing START switch on CTS.

b. Setting up and executing jobstream.

Specific setup instructions presume that the user has communicated the proper hardware setup of data tapes, scratch tapes and/or scratch MLU to the operator.

b. Setting up and executing jobstream (continued)

Step Action

1. If required, load and ready commercial tape transports.

2. If required, prepare the MLU for writing and activate the ON LINE and WRITE ENABLE switches on the ARMM, both must be illuminated. The MLU must be positioned at BOT.

3. Load and ready the card reader with the jobstream deck. Press END OF FILE and START, wait for the READY light on card reader. If less than 3 cards, put 2 blank cards behind last card.

4. At this point the jobstream is ready for execution. If PUP was just loaded, press START. If PUP just completed a previous major job (DIG = 770000), press START. If terminating a previous bad run (computer stopped with error DIG lights) press MASTER RESET.

   NOTE: Should PUP fail to start on pressing the START switch, press MASTER RESET.

c. Diagnostic displays and recovery conditions.

The DIG display values and recovery procedures are given in Section 2, Table 2-I.

SECTION 2
GENERATION OF PLANIT COMMERCIAL OR MLU LOAD TAPES

2.1 General Information

PUP allows the user to prepare PLANIT commercial or MLU load tapes by the use of various control cards and program decks. Patches may be inserted into programs and each boot deck will be written on the system tape.

2.2 User Information - Data Preparation

Each program record to be written on commercial tape or MLU is first prepared as an input consisting of a //B or O card, a program deck, some number of patch cards, and one blank card. The following paragraphs describe the various components for a single program (see Figure 2-2 and Appendix D for examples).

a. Output Media Card

This card should precede other control cards to define the output media for the load tape:

//MXY specifies output media for master load tape

X = A for ARMM

= C for Commercial tape

Y = Commercial tape 0-3

= Blank for ARMM

If this card is not present the output media is assumed to be an ARMM.

b. Program Type Card

Program type cards preceed the program and may be any of the following:

1. //BB - Write a bootable record that is in bootable format.
2. //BO - Write a bootable record that is in object format.
3. //OB - Write an object record that is in bootable format.
4. //OO - Write an object record that is in object format.

c. Program Deck

Object program decks are generated by the compiler. Bootable program decks are generated by PUP (see Section 3).

d. Patch Cards

Patch cards have the following format:

1. Control card: columns 1-3 are '//P'.

2. Patch card (see Figure 2-1 for example)
   a) Columns 1-4 = address of patch
   b) Column 5 = blank
   c) Starting in column 6 are either 4 digit hex halfword patches or 5 digit instruction patches; each halfword patch is separated by a comma (,); the first blank terminates the patch. If more than one halfword patch is put on the same card, each halfword will be assigned consecutive addresses starting with the address on the patch card.

e. Blank card.

A single blank card is placed behind the last punched card of the program deck or the last patch card.

2.3 Operating Procedure

To generate a tape the following operational steps are required:

a. The hardware must be setup and the PUP program loaded as given in Section 1.5.a.

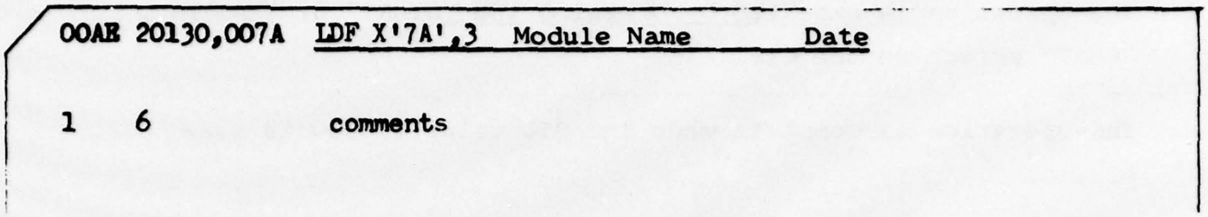
b. The input jobstream consisting of one or more program decks and followed by a '// ' end card is prepared and the device(s) readied. If the jobstream is on commercial tape X, then place a '//TX' card in the card reader.

c. The output device (commercial tape or MLU) is prepared for writing and positioned at BOT (beginning of tape).

d. The operation is executed by pressing the "START" or "MASTER RESET" switch on the CTS.

e. The operation is complete when the DIG value 770000 is displayed.

f. Table 2-I shows the DIG values used during the execution of the program. Most errors are not recoverable and require restart of the entire operation.



OOAB 20130,007A LDF X'7A',3 Module Name Date

1 6 comments

Figure 2-1. Sample Patch Card.

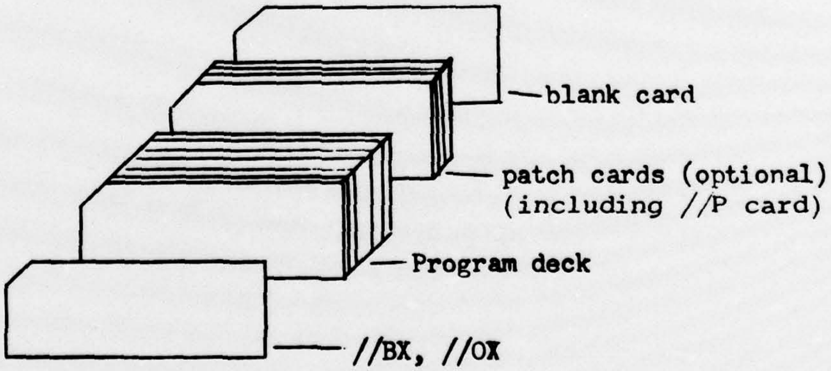


Figure 2-2. Sample Program Deck Set-up.

TABLE 2-I . PUP Generation DIG Values

DIG VALUE	MEANING	RECOVERY
100XX0	Input in progress on device XX . (See Note)	Not applicable.
107XX0	End of tape reached reading MLU on XX.	Mount new TTC, press 'COMPUTER INTERRUPT' on IOU.
170XX0	Input I/O error on device XX.	If XX=14 (card reader) reload card reader with last card in stacker 2, press 'START' on CTS. If XX=21 (MLU), press 'START' on CTS. If XX=other, recovery not possible.
171XX0	Device time-out on input device XX.	Ready device XX, press 'START' on CTS.
200XX0	Output in progress on device XX.	Not applicable.
207XX0	End of tape reached writing device XX.	Mount next reel, press 'COMPUTER INTERRUPT' on IOU.
270XX0	Output I/O error on device XX.	If XX=15 (card punch), press 'START' on CTS - bad card repunched. Job continues. If XX=21 (MLU), press 'START' on CTS. If XX=other, recovery not possible.
271XX0	Device time-out on output device XX.	Ready device XX, press 'START' on CTS.
300000	No type 1 card in compiler object deck.	Recovery not possible. Press 'START' on CTS to flush this deck and continue next job.
310000	Checksum error reading compiler object deck.	Recovery not possible. Press 'START' on CTS to flush this deck and continue next job.
320000	No type 9 card in compiler object deck.	Recovery not possible. Press 'START' on CTS to flush this deck and continue next job.
330000	Illegal card after type 9.	Recovery not possible.Press 'START' on CTS to flush this deck and continue next job.

TABLE 2-I. PUP Generation DIG Values (Continued)

DIG VALUE	MEANING	RECOVERY
340000	Compiler object deck is out of sequence.	Recovery not possible. Press 'START' on CTS to flush this deck and continue next job.
770000	'//' card reached, end of job.	Not applicable. Press 'START' on CTS to continue to next job.
777777	Level 2 hardware error.	Recovery not possible. Major hardware malfunction.

Note: XX = 10 for Commercial Tape Unit 0
11 for Commercial Tape Unit 1
12 for Commercial Tape Unit 2
13 for Commercial Tape Unit 3
14 for Card Reader
15 for Card Punch
16 for High Speed Printer (HSP)
17 for Selectric Typewriter
21 for MLU

SECTION 3
GENERATION OF BOOTABLE MEDIA (DECKS, TAPE) FROM COMPILER OBJECT OUTPUT

3.1 General Information

PUP provides the capability to convert compiler object output to a format that is bootable directly into the computer. This operation must be performed whenever the PLANIT Operating System is recompiled as this program is the first module on the PLANIT system tape and is bootstrapped in to start the system load process.

3.2 User Information - Data Preparation

Data preparation for the bootable format generation process is the same as any other job stream set-up. The proper control cards in conjunction with the object deck to be converted are put together and executed. The following paragraphs describe the components necessary to generate a bootable record. See Appendix D for an example.

a. //BO Card

This card tells PUP that a bootable record is to be generated and that the input is in compiler object format. The output will go to the tape specified in the //MXY card. (See Appendix A) If punched output is desired, use a '//CO' card (b. below) instead of the //BO.

b. //CO Card

This card tells PUP that a bootable card deck is to be generated and that the input is in compiler object format. The deck that is punched will have the Type 1 card from the object input duplicated at the front and the Type 9 card last. These two cards should be removed before actual use of this deck. They are only supplied to provide a check that the punched deck is correct.

c. Object Deck

The complete object deck as output from the compiler is used. If no Type 1 Card is included with the deck the program will halt and output an appropriate DIG code (see Table 2-I). The object deck will be checked by

c. Object Deck (Continued)

check summing during read-in and if an error is detected, the program will halt and output an appropriate DIG code (see Tablé 2-I). If no Type 9 Card is included with the deck, the program will halt and output an appropriate DIG code (see Table 2-I).

d. //P Card

This card is optional. If included, it indicates that the following cards are patches to be applied to the program deck just read in.

e. Blank Card

3.3 Operating Procedure

To generate a bootable media program from compiler object output, the following operational steps are required:

a. The hardware must be set up and the PUP program loaded as described in Section 1.5.a.

b. The input stream consisting of one or more deck setups as described in 3.2 above and followed by a '//' end card is prepared and made ready in the card reader.

c. The card punch is made ready (if required).

d. The operation is executed by pressing 'START' or !MASTER RESET' switch on the CTS.

e. The operation is complete when the DIG value 770000 is displayed.

f. Table 2-I shows the DIG values used during the execution of the program. Most errors are not recoverable and require restart of the entire operation.

SECTION 4

PROCESS SPECIAL PLANIT DATA

4.1 General Information

PUP allows the user to process two forms of special PLANIT data: input data and output data. This function is applicable to operation from the MLU or commercial tape drives only. The function will prepare input tapes to be read by the PLANIT system or convert output data from PLANIT to a format usable for off-line data processing. The input data is either PLANIT lesson material or the PLANIT 'CARDS FILE'. The output data is student record information written by PLANIT.

4.2 User Information - Data Preparation

Data preparation to process special PLANIT data depends on whether input or output data are to be processed and the devices to be used for processing.

One basic control card is used for all processing with a number of optional/variable fields that are used depending on processing conditions. The control card is '//DXYZ' where the XYZ fields are variable depending on application. The options possible are described in the following paragraphs.

4.2.1 INPUT DATA Processing

Special PLANIT input data can be input to PUP either on commercial tape or punched cards. If commercial tape is used this step assumes the tape is in card image unblocked records format. All data in EBCDIC characters.
The data will be output in 8000 character records in ASCII character set. All trailing blanks on input card images will be deleted. This data will be output to either a commercial tape or MLU for later use. If a commercial tape is specified, a tape density control card (//DN08, //DN16) must precede the data to tell PUP which density to use in writing the output. A tape that will be used on a Potter unit must be 800 BPI. If the density control card is not present 1600 BPI is assumed.

The input processing control card is as follows:

'//DIYZ', where 'Y' and 'Z' are defined below:

Y = 0-3 for commercial tape 0-3 input
= blank for punched card input

Z = 0-3 for commercial tape output
= blank for MLU output.

4.2.2 OUTPUT DATA Processing

Special PLANIT output data can be input to PUP either on commercial tape or MLU.

The data is expected to be in 8000 character records in ASCII character set. The data will be deblocked into 80 column card images in EBCDIC character format. This data will be output to either commercial tape or punched cards.

The output processing control card is as follows:

'//DOYZ', where 'Y' and 'Z' are defined below:

Y = 0-3 for commercial tape 0-3 input
= blank for MLU input

Z = 0-3 for commercial tape 0-3 output
= blank for punched card output.

4.3 Operating Procedure

To process special PLANIT data the following operational steps are required:

a. The hardware must be set up and the PUP program loaded as described in Section 1.5.a.

b. The input stream as described in 4.2 above is prepared and made ready in the card reader.

c. The card punch is made ready if cards are to be punched.

d. Input tape is mounted and made ready on appropriate drive if tape input is specified.

e. A scratch tape is mounted and made ready if tape output is specified.

f. The operation is executed by pressing 'START' or 'MASTER RESET' switch on the CTS.

g. The operation is complete when the DIG value '770000' is displayed.

h. Table 2-I shows the DIG values used during the execution of the program. Most errors are not reconverable and require restart of the entire system.

SECTION 5

UPDATING THE PLANIT OBJECT LIBRARY TAPE

5.1 General Information

PUP allows the user to update the PLANIT Object Library Tape by deleting, replacing and/or adding programs while copying the original library tape. The library tape is in card image format and serves as the jobstream input to generate the PLANIT system tape as described in Section 2 and Appendix D.

Because of the critical nature of maintaining a perfect library tape this chapter includes paragraph 5.3 which deals with the history, development, peculiarities and trouble-shcoting techniques of the library tape. The paragraph also deals with two not so obvious procedures, copy library tape and initial generation of the library tape.

5.2 User Information - Data Preparation

Data preparation for the library update program depends largely on the modifications to be made to the original library tape. The following paragraphs describe the jobstream (and data) input necessary to update a library tape as well as the cards required to summarize the new library tape (Section 6) and generate a PLANIT system tape (Section 2). Figure 5-1 and Appendix D illustrate sample update / summarize / generate jobstreams.

a. //AMXN Control Card

This card causes the original library tape mounted on commercial tape transport X (0, 1, 2 or 3) to be rewound to load point. The original library tape is copied to the scratch tape mounted on commercial tape transport N (0, 1, 2 or 3). During the copy process changes (adds, deletes, and replaces) are performed according to card reader input.

b. Replace, Add, Delete Control Cards

Each program on library tape which is to be modified requires one of the following control cards:

(1) //ARXX - Replace program XX on the original library tape with the program in the card reader.

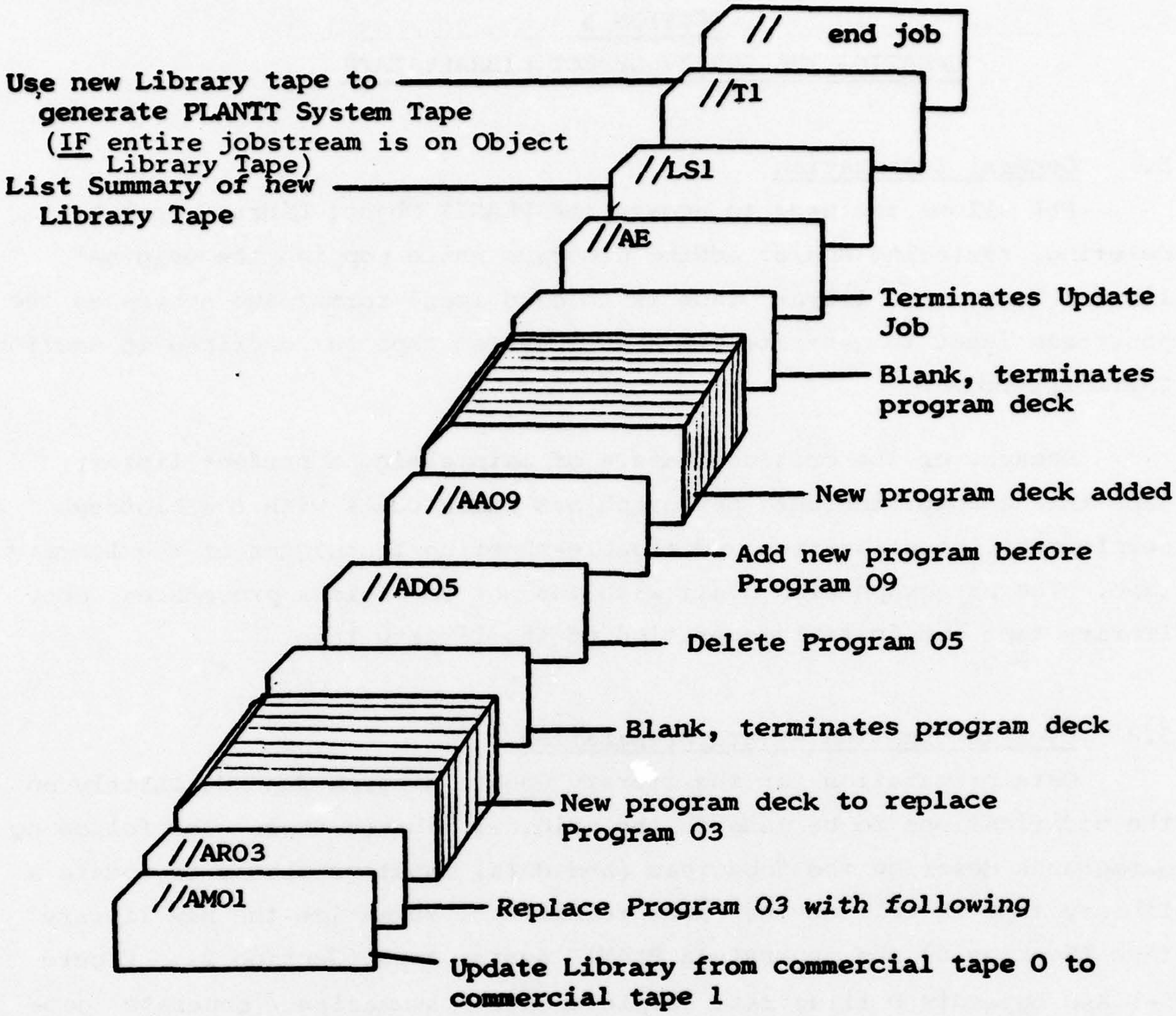

// end job
//T1
//LS1
//AE
//AA09
//AD05
//AR03
//AM01
Use new Library tape to generate PLANIT System Tape (IF entire jobstream is on Object Library Tape)
List Summary of new Library Tape
Terminates Update Job
Blank, terminates program deck
New program deck added
Add new program before Program 09
Delete Program 05
Blank, terminates program deck
New program deck to replace Program 03
Replace Program 03 with following
Update Library from commercial tape 0 to commercial tape 1

Figure 5-1 Sample Update Library Job Deck

(2) //AAXX - Add the program in the card reader before program XX on the original library tape.

(3) //ADXX - Delete program XX on the original library tape.

The value XX is a decimal program number with 00 being the first program on the original library tape. (See Section 5.3, Table 5-I.) A program on either the library tape or in the card reader is defined as a collection of 0 to n non-blank card sets terminated by one and only one blank card.

The replace (with program), add (with program) and delete control cards are assembled in ascending order according to the program number XX and placed behind the //AMXN card.

c. //AE Control Card

This card indicates that no more modifications to the library tape are to be made and causes the update program to copy the balance of the original library tape. The copy is complete when the // end card is encountered and copied. Both tapes are then rewound and the update routine returns control to PUP.

d. //LSX Control Card (Optional)

This card causes a summary of the new library tape to be made. X(0, 1, 2 or 3) is the commercial tape unit where the new library tape is located. See Section 6 for additional details.

e. //TX Control Card (Optional)

This card causes the new library tape mounted on commercial tape unit X (0, 1, 2 or 3) to be used as the input jobstream to generate a PLANIT system tape. See Section 2 for details.

Note: A computer stop with DIG display 770000 occurs after the update step, after the summarize step and after the generate step. This allows the summary reports to be checked for errors prior to writing on the system tape. See Table 2-I.

5.3 Library Tape Trouble-Shooting Techniques

This section deals with the peculiarities of maintaining a valid library tape. An understanding of these items enables the user to interpret the possible diagnostic messages generated by the summarize step (Section 6) in a more straightforward manner.

The following paragraphs represent the more significant concepts and trouble-shooting techniques "discovered" during the development of the two routines, update and summarize, and the creation of the present library tapes. The concepts and techniques presented are derivable from other sections of this manual but the following presentations are useful.

(1) Initial Generation of the Library Tape

The initial generation library tape can be made with an off-line card-to-tape. It can also be made with the facilities of the update routine. PUP is loaded and an update operation started (//AMXN). Both the input and output tapes are scratch tapes. The output tape is built by using //AAOO cards in front of each program deck starting with PUP and ending with the following cards:

```
//AAOO
//     (end of job)
(blank card)
```

The empty card reader after the last blank card has been read will cause a card reader fault but the output tape is already the initial library tape.

(2) Copy Library Tape

A backup copy of the library tape may be made using two cards:

```
//AMXN
//AE
```

This is an update step with no actual changes.

(3) Jobstream Input

The entire library tape is used as jobstream input. The PUP boot deck card images are ignored by PUP as unrecognized control cards. The first card to cause action is the first recognized PUP control card following the PUP boot deck.

(4) PUP Boot Deck Copied

The library update routine begins by rewinding the original library tape so that it can also copy the PUP boot deck. The boot load will have read part of the tape and thus a manual rewind is not necessary.

(5) More Than One Blank Card

Occasionally a library tape may exist with 2 or more blanks in a row as indicated by the summary report. Update can be used to remove the blanks. Update considers the second blank card to be the termination card of a zero card program. By deleting that program the terminating blank is also deleted, for example:

If the summary report has the following message for program 5: (See Table 5-I):

```
FAULT  **  3 BLANKS, 1 REQUIRED
```

Then the following delete cards are prepared for the update step:

```
//AD05
//AD06
```

The next item will make this relationship clearer.

(6) Calculating Update Program Numbers

The update program numbers can be calculated from the summary report "TAPE POS" number with the aid of the following formula:

$N = T - 1 + E$ where:

N = update program number

T = TAPE POS number from summary report

E = sum of the extra blanks before the Tth program. Extra blanks are reported as fault messages, i.e., "FAULT ** 3 BLANKS, 1 REQUIRED" is two extra blanks.

5.3 Library Tape Trouble-Shooting Techniques (Continued)

(7) //R Card in Library

A stand alone //R card in the library is a valid entry treated as a separate program and has some useful function. The comment section columns 5-80 should be used to describe its purpose such as "POS TEST PROGRAM FROM CARD READER". The //R card is printed as the program title in the summary report. The //R card should be followed by a blank card so that it can be updated at a later date. It can be ignored by using two //TX cards during the generate step instead of one.

Table 5-I. Sample Extra Blank Fault

UPDATE PROGRAM NUMBER	SUMMARY REPORT OUTPUT (parenthetical comments are not printed)
00	1 **** NO PROGRAM CARD FOUND (PUP)
01	2 //B . . . .
02	3 //M . . . .
03	4 //M . . . .
04	5 //B . . . .
	FAULT ** 3 BLANKS, 1 REQUIRED
05	(1st extra blank's zero length program)
06	(2nd extra blank's zero length program)
07	6 //O . . . .
08	7 //O . . . .

5.4 Operating Procedure

To perform a library update job the following operational steps are required:

a. The hardware must be setup and the PUP program loaded as given in Section 1.5.a.

b. The input jobstream (see Figure 5-1 for a sample) is loaded in the card reader.

Note: Only the card reader is permitted as the jobstream input for a library update job.

c. Load and ready an output tape on the correct commercial tape transport.

d. If not already mounted, load and ready the original library tape on the correct commercial tape transport.

e. The operation is executed by pressing the "START" or "MASTER RESET" switch on the CTS.

f. The operation is complete when the DIG value 770000 is displayed.

g. Table 2-I shows the DIG values used during the execution of the program. Most errors are not recoverable and require restart of the entire operation.

SECTION 6
SUMMARIZING THE PLANIT OBJECT LIBRARY TAPE

6.1 General Information

PUP allows the user to obtain a printed summary of the PLANIT OBJECT Library Tape. The summarize routine is normally run after updating the library tape (Section 5) but may be run at any time. The summarize routine produces two reports, each with diagnostic messages to flag detected potential and actual faults. The first report is primarily a list of each PUP control card used to create a system tape in order of their occurrence. The second report is an index of contents of the library tape showing key statistics for each program.

6.2 User Information - Data Preparation

The summarize routine requires only one control card to cause execution, the format of that card is as follows:

Columns 1-5. '//LSX', where X is the commercial tape unit 0, 1, 2 or 3 where the PLANIT OBJECT Library Tape is mounted.

Columns 9-64. Will be printed as title on each page of summary.

Appendix B is a sample summarize output listing. Section 6.4 explains the output and describes the possible diagnostic messages.

6.3 Operating Procedure

To summarize the PLANIT OBJECT Library Tape, the following operational steps are required:

a. The hardware must be setup and the PUP program loaded as given in Section 1.5.a.

b. The PLANIT OBJECT Library Tape is mounted and made ready on commercial tape unit X (X = 0, 1, 2 or 3).
NOTE: Tape X is rewound by the routine before starting and at completion.

c. The //LSX control card is loaded into the card reader jobstream.

d. The line printer is made ready.

e. The operation is executed by pressing the "START" or "MASTER RESET" switch on the CTS.

f. The operation is complete when the DIG value 770000 is displayed.

g. Table 2-I shows the DIG values used during the execution of the program. Most errors are not recoverable and require restart of the entire operation.
NOTE: Section 5 shows the summarize routine used in conjunction with updating the library tape.

6.4 Summarize Output Explanation

Appendix B is a sample of the two reports produced by the summarize routine. The diagnostic messages printed refer to violations of various "rules" in Sections 1, 2 and 5 and may affect the jobstream control, the generation of a system tape or the library update routine. A valid PLANIT OBJECT Library Tape should have no warning or fault messages depending on the approach taken in structuring the object library. The following information describes the two reports (printer output is underlined):

A. First printed report

1. **** //CARD LISTING ****
heading.

2. Normal data.

a. All PUP control cards (each beginning with //) are listed in the order of their occurrence.

b. NNNN CARD IMAGES FOLLOWED BY NNNN BLANK CARD(S)
This message is printed upon the detection of a blank card or cards. The NNNN characters are replaced by the actual values in each case.

The card image count is the number of program records following the //BX, //OX until a blank card is detected.

The blank count is the number of consecutive blank cards and terminates a program deck.

NOTE: The first program on tape is presumed to be the PUP load deck and all records from the beginning of the tape to the first blank card are considered program cards of PUP.

3. Diagnostic messages. The following warning messages are printed to the right of the card image to which they apply except "e." which requires a full line:

   a. WARNING GOES TO CARD READER

   This message is printed after an //R card as a reminder to the user/operator that card reader input is expected when building a System tape.

   b. WARNING ILLEGAL //CARD

   This message is printed after any unrecognized PUP control card not normally encountered in building a system tape. Fault message B.3.c (page 6-5) will also be printed.

   c. WARNING // CARD IN PROGRAM, RECORD NNNN

   This message is printed when '//' is found in columns 1-2 of any card in the program deck. A program deck is PUP (see note with A.2.b) or the cards following a //BX or //OX card and before a blank card. NNNN is the record number of the card starting with 1. The card in question could be a valid program card or a misplaced PUP control card, inspect listing to determine.

6.4 Summarize Output Explanation (Continued)

d. WARNING XXX NON // CARD BEFORE BLK OR PRG CARD

This message is printed when a non // card is found when one was expected. Only the first four are listed and XXX will be replaced with 1ST, 2ND, 3RD or 4TH. More than 4 such cards cause message A.3.e (below) to be printed. Fault message B.3.d (page 6-5) will also be printed.

e. ** WARNING NNNN NON // CARDS FOUND BEFORE NEXT CARD OR BLANK (ONLY 1ST FOUR ARE PRINTED, SEE ABOVE WARNINGS)

(Above printed as one (1) line.)

A // card was expected to be read. Instead NNNN consecutive non // cards were read. NNNN count does not include the 1st four cards which were printed (see d.above).

f. WARNING MORE THAN ONE BLANK FOLLOWS PROGRAM

This message is printed when more than 1 blank terminates a program. The extra blanks must be removed. Fault message B.3.b (page 6-5) will also be printed.

B. Second printed report

1. Column headings:

a. TAPE POS - This is the position on the tape starting from 1.

b. PROGRAM CARD - This is the card image of any of the key cards used as a program title card. Key cards are //BX, //MX, //R or // cards.

c. PATCHES - The count of patch cards for a program.

d. RECORDS - The count of the program card images for a program.

e. BSL DATA - The boot bytes from the first card of the program if it was a boot deck.

2. Normal data.

The normal data is as indicated in 1. above.

NOTE: The first item on the PLANIT OBJECT Library Tape is the PUP boot deck. It is given the following dummy title:

**** NO PROGRAM CARD FOUND (PUP ?)

3. Diagnostic messages. The following fault messages are printed as separate lines following the program entry to which they apply.

a. NOT USED

b. FAULT ** NNNN BLANKS, 1 REQUIRED

Each program must be followed by only 1 blank, also see warning message in 1st listing. One blank only is required for proper operation of the Library update routine.

c. FAULT ** NNNN ILLEGAL // CARDS

Count of illegal // cards found, see warning messages in 1st listing.

d. FAULT ** NNNN TOTAL NON // CARDS

The total of non // cards read when one was expected, see warning messages in 1st listing.

6.4 Summarize Output Explanation (Continued)

e. FAULT ** NNNN =BSL EL RECORDS EXPECTED

The number of records NNNN calculated from the boot strap load data on the 1st card of the program does not agree with the number of program cards read. Something is wrong with the program boot deck which must be corrected.

f. FAULT ** 1 BAD BSL, NO 2FDO

The check character and complement (2FDO) was not found on the 1st card of the program. Something is wrong with the program boot deck which must be corrected.

NOTE: The MADCAP EOF record which is written as an object program but does not have the 2FDO bytes will not produce this fault message.

g. NOT USED

h. FAULT ** NNNN // END NOT SEGREGATED

The card marking the end of tape (//) was not preceeded by a blank card. A blank must preceed the // end card for proper operation of the library update routine.

i. FAULT ** 0 NO PROGRAM

A missing control card (//BX or //OX) or no program deck causes the program record count to be 0. Inspect to determine.

4. Default data.

The following default data is entered for each program for which no //BX, //OX or R card is found:

a. Program card: **** NO PROGRAM CARD FOUND

b. BSL data: FFFF FFFF FFFF FF

6.4 Summarize Output Explanation (Continued)

5. Last line.

The last line printed gives the total number of warning and fault messages printed in the two reports. Normal usage of the PLANIT OBJECT Library Tape should result in no warnings or faults depending on the approach taken in building the object library tape (see Appendices B and D). The format of the message is as follows:

```
XXXX WARNINGS   YYY FAULTS
```

SECTION 7
PUNCH PLANIT TRANSLATION TAPE

7.1 General Information

PUP provides the capability to punch the translated PLANIT source tape to create card input for the TACPOL 'B' Compiler. This capability is required because the translated PLANIT source is in EBCDIC character format and the compiler will only accept cards in EBCDIC format, tapes must be in ASCII (see Section 8).

7.2 User Information - Data Preparation

Data preparation for this function only requires the creation of the proper control cards to initiate and terminate the job. The following paragraphs describe the control cards necessary to perform this function.

a. //PPX Card

This card initiates the function and tells the number of the tape drive containing the input tape. The tape number is field X (col 5) in the control card.

b. // Card

This is the standard card to terminate the job stream and rewind all tapes.

This function assumes the input tape is blocked 80X10 and is terminated by a tape mark. Punching will continue until the tape mark is encountered, at which time the job will terminate and rewind the input tape.

7.3 Operating Procedure

To generate the punched PLANIT source the following operational steps are required:

a. The hardware must be set up and the PUP program loaded as described in Section 1.5.a.

b. The input stream as described in 7.2 above is prepared and made ready in the card reader.

c. The card punch is made ready.

d. The operation is executed by pressing 'START' or 'MASTER RESET' switch on the CTS.

e. The operation is complete when the DIG value '770000' is displayed.

f. Table 2-I shows the DIG values used during the execution of the program. Most errors are not recoverable and require restart of the entire operation.

SECTION 8

CREATE LIBRARIAN INPUT TAPE FROM PLANIT TRANSLATION SOURCE TAPE

8.1 General Information

PUP provides the capability to generate a tape suitable for L3050 librarian input from the translated PLANIT source tape. This function is necessary because the translated source tape is unlabeled and in EBCDIC character format and the L3050 librarian requires labeled ASCII input.

8.2 User Information-Data Preparation

This function assumes the input tape is blocked 80X10 in EBCDIC character format. The 'CARDS FILE' must be the last data on the tape as the string of dollar signs ($$$. . .) at the end of 'CARDS' is used to terminate tape processing.

The output tape that is created will be labeled, unblocked, in ASCII character format. The format of the tape is shown in Figure 8-1. Each HDR1 and EOF1 contains the name of the file. The files are named sequentially from 'FILE01' to "FILEnn' where nn is the last file number. The volumn label created is number 0000.

Data preparation for this function requires the creation of the proper control cards to initiate and terminate the job. The following paragraphs describe the control cards necessary to perform this function.

a. //PCXY Card

This card initiates the function and tells PUP the input and output tape drive numbers. The number (0-3) in column 5 (X) is the input tape number. The number (0-3) in column 6 (Y) is the output tape number.

b. // Card

This is the standard control card to terminate the job stream and rewind all tapes.

8.3 Operating Procedure

To generate the librarian input tape the following operational steps are required:

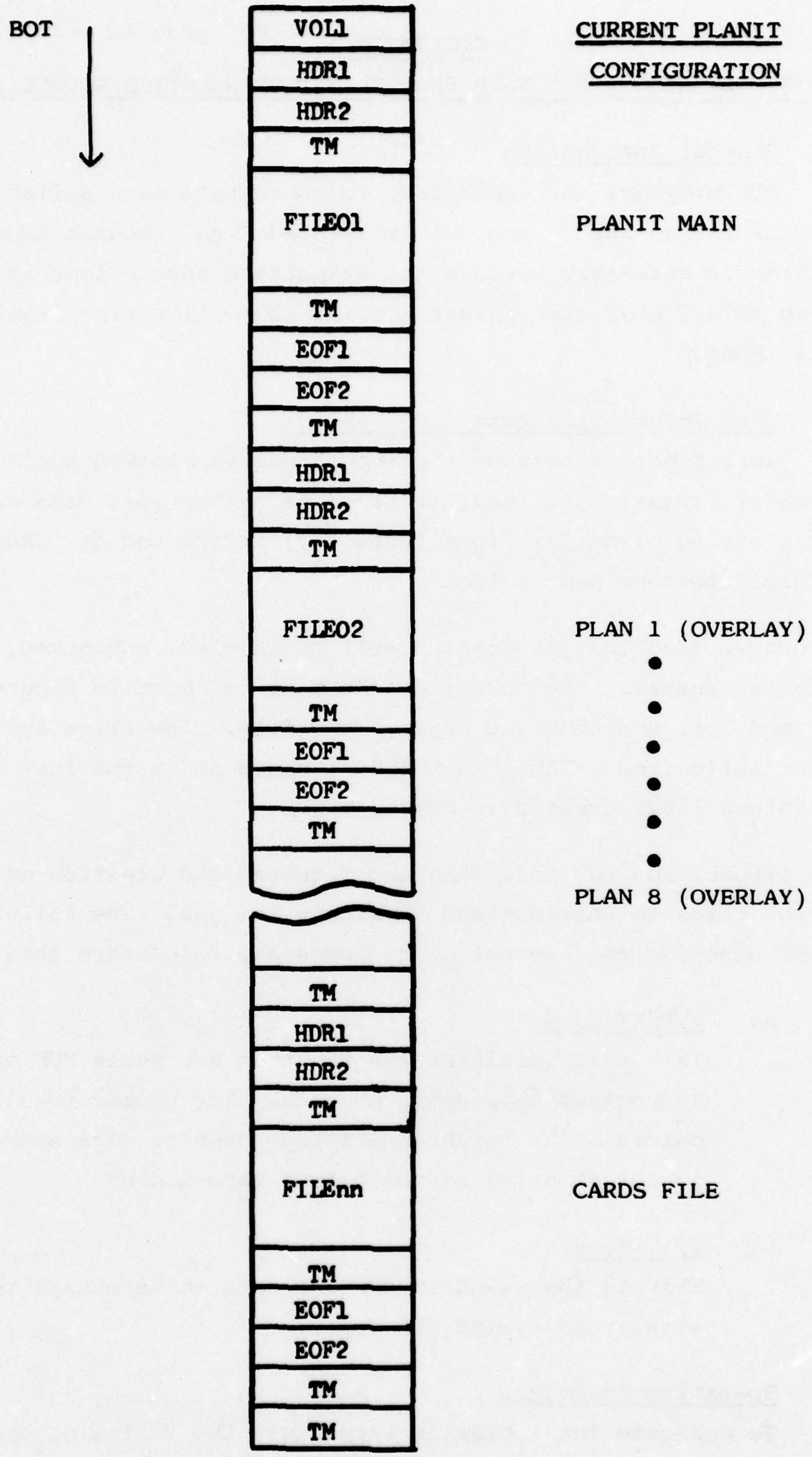


Figure 8- 1. Volume Organization - Librarian Input Tape

a. The hardware must be set up and the PUP program loaded as described in Section 1.5.a.

b. The input stream as described in 8.2 above is prepared and made ready in the card reader.

c. The operation is executed by pressing 'START' or 'MASTER RESET' switch on the CTS.

d. The operation is complete when the DIG value '770000' is displayed.

e. Table 2-I shows the DIG values used during the execution of the program. Most errors are not recoverable and require restart of the entire operation.

SECTION 9
GENERAL CHARACTER CONVERSION ROUTINE

9.1 General Information

PUP provides the capability to perform character conversion on various data. This function will convert the characters in the input data stream and create a new converted output. Conversion can be from any character set to any character set; the conversion criteria is specified as input to the function. The input and output media may be either cards or tape. The converted output will be printed as part of the function.

9.2 User Information-Data Preparation

Data preparation for this function requires creating the necessary control cards and data conversion information. The following paragraphs describe the control cards and conversion information necessary to perform this function.

a. //CVXY Card

This card tells PUP that the data conversion function is to be performed. Column 5 ('X') specifies the input media for the data to be converted. A blank denotes card input and a number (0-3) indicates the tape unit containing the input. If tape input is used the tape must be card images unblocked format (80X1). Column 6 ('Y') specifies the output media for the converted data. A blank denotes card output and a number (0-3) indicates the tape unit to receive the data. If tape is specified, the tape will be written in card image unblocked format (80xl).

b. IN Card

This card specifies the input characters which are to be converted. Columns 1 and 2 must contain the characters 'IN' with the remaining 78 columns available for character data. If less than 78 characters are specified, a 12-11-0-7-8-9 (hex FF) must follow the last valid character. This card must follow the //CV card or an error will be indicated and the job must be restarted. Any character not on this card will be output unconverted, i. e., only character differences need be included (See Figure 9-1 and Table 9-I).

c. OT Card

This card specifies the characters that output is to be converted to. Columns 1 and 2 must contain the characters 'OT' with the remaining 78 columns available for character data. The conversion characters on this card must have a one to one correspondence with those on the 'IN' card for proper conversion to take place. This card must follow the 'IN' card (b above) or an error will be indicated and the job must be restarted.

d. Data Deck

The data deck to be converted must follow the 'OT' card if card input is specified in the //CV card.

e. // Card

This is the standard card to terminate the job stream and rewind all tapes.

9.3 Operating Procedure

To perform the data conversion function the following operational steps are required:

a. The hardware must be set up and the PUP program loaded as described in Section 1.5.a.

b. The input stream as described in 9.2 above is prepared and made ready in the card reader.

c. The card punch is made ready if cards are to be punched.

d. Input tape is mounted and made ready on appropriate drive if tape input is specified.

e. A scratch tape is mounted and made ready if tape output is specified.

f. The operation is executed by pressing 'START' or 'MASTER RESET' switch on the CTS.

g. **The operation is complete when the DIG value '770000' is displayed.**

h. **Table 2-I shows the DIG values used during the execution of the program. Most errors are not reconverable and require restart of the entire system.**

9.4 AN/GYK-12 PLANIT System Character Sets

The AN/GYK-12 PLANIT character set includes the letters A through A, zero (0) through 9 and the special characters shown in Table 9-I. The AN/GYK-12 PLANIT system operates internally with the ASCII Character Set and code converts to/from the EBCDIC Character Set for output to or input from the commercial card punch, card reader and the high speed printer.

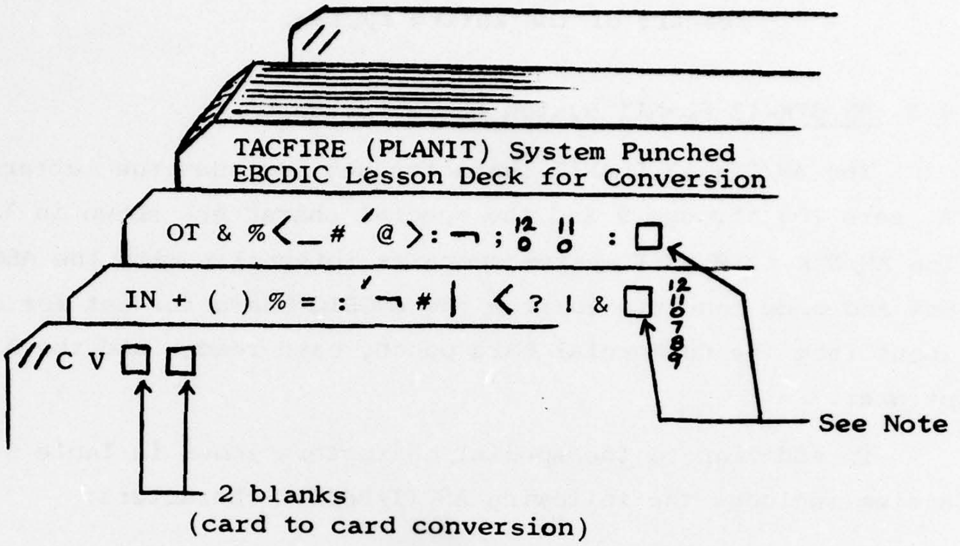
In addition to the special characters shown in Table 9-I, the system includes the following ASCII/EBCDIC characters:

Quote (") - EBCDIC punch 7,8
At (@) - EBCDIC punch 4,8
Greater than (>) - EBCDIC punch 0,6,8
Underscore (_) - EBCDIC punch 0,5,8

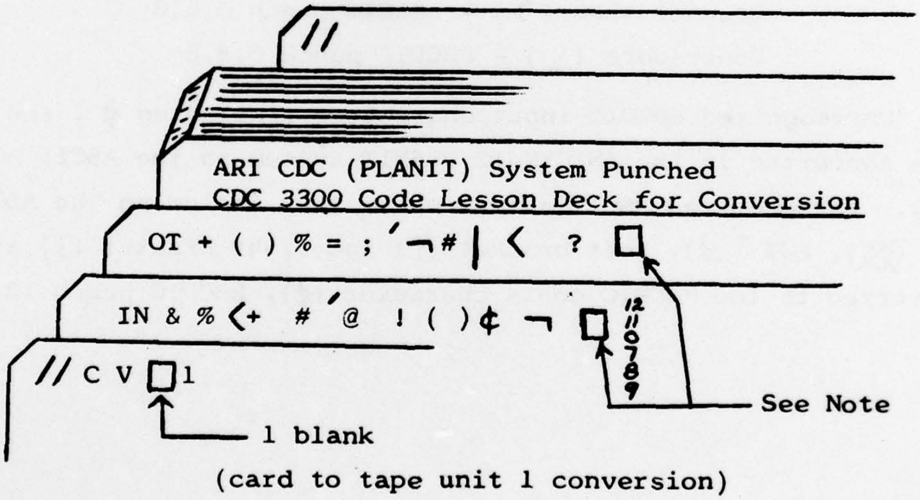
Unrecognized EBCDIC input characters (including ¢ , see below) are code converted in the AN/GYK-12 PLANIT system to the ASCII ACK character (⋈). Unrecognized ASCII output characters including the ASCII ACK (⋈), NAK (⊗), EOT (⊣), left bracket ([) and right bracket (]) are code converted to the EBCDIC cents character (¢); EBCDIC punch 12,2,8.

Figure 9-1. Typical Job Deck for PLANIT Lesson Character Conversion

a. TACFIRE SYSTEM (EBCDIC) Lesson to ARI UNIVAC 1108 Code (Using 029 Key Punched Job Cards)


//
TACFIRE (PLANIT) System Punched
EBCDIC Lesson Deck for Conversion
OT & %<_# ´@>:¬; 12 0 11 0 : □
IN + ( ) % = : ´¬#| <?!& □
12 11 0 7 8 9
//C V □ □
See Note
2 blanks
(card to card conversion)

b. ARI CDC 3300 to TACFIRE System (EBCDIC) Code (Using 029 Key Punched Job Cards)


//
ARI CDC (PLANIT) System Punched
CDC 3300 Code Lesson Deck for Conversion
OT + ( ) % = ; ´¬#| < ? □
IN & %<+ # ´@ ! ( ) ¢ ¬ □
12 11 0 7 8 9
//C V □1
See Note
1 blank
(card to tape unit 1 conversion)

Note: Leave last character position blank (if blank converts to blank). This speeds up conversion since this is the first character in the table lookup and blank is the most frequent character to occur.

Table 9-I

PLANIT Character Set Differences - AN/GYK-12 and ARI Commercial Systems

TACFIRE PLANIT (ASCII) CHARACTER	IBM (EBCDIC) PUNCH	029 KEY PUNCH CHARACTER	CDC PUNCH	029 KEY PUNCH CHARACTER	UNIVAC PUNCH	029 KEY PUNCH CHARACTER
Plus +	12,6,8	+	12	&	12	&
Minus -	11	-	11	-	11	-
Asterisk *	11,4.8	*	11,4,8	*	11,4,8	*
Slash /	0,1	/	0,1	/	0,1	/
Open Paren (	12,5,8	(	0,4,8	%	0,4,8	%
Close Paren)	11,5,8	)	12,4.8	<	12,4,8	<
Period .	12,3,8	.	12,3,8	.	12,3,8	.
Percent %	0,4,8	%	12,6,8	+	0,5,8	—
Equals =	6,8	=	3,8	#	3,8	#
Comma ,	0,3,8	,	0,3,8	,	0,3,8	,
Colon :	2,8	:	2,8	:	5,8	'
Semi-colon ;	11,6,8	;	5,8	'	11,6,8	;
Prime '	5,8	'	4,8	@	4,8	@
Backslash \	11,7,8	¬	11,2,8	!	0,6,8	>
Blank		(Blank)		(Blank)		(Blank)
Pound Sign #	3,8	#	12,5,8	(	2,8 (1)	:
Ver Arrow ^	12,7,8	|	11,5,8	)	11,7,8	¬
Hor Arrow <	12,4,8	<	12,2,8	¢	11,6,8	;
Question Mk ?	0,7,8	?	11,7,8	¬	12,0 (multi-punch)	$\overset{12}{0}$
Dollar Sign $	11,3,8	$	11,3,8	$	11,3,8	$
Exclamation !	11,2,8	!	None	(Blank)	11,0 (multi-punch)	$\overset{11}{0}$
Ampersand &	12	&	None	(Blank)	2,8	:

(1) An ampersand is used for the Pound Sign in this system.

(2) Letters A-Z and numbers 0-9 are same in all three code sets.

APPENDIX A
PUP CONTROL CARDS

This appendix contains the format, a brief description and section references for all PUP control cards.

TABLE A-I. PUP CONTROL CARDS

(Note 1) CONTROLLING CHARACTERS	MEANING	REFERENCE MANUAL SECTION
//TX	Assign commercial tape X as input device.	1.4.a
//R	Assign card reader as input device.	1.4.b
//	Marks end of jobstream.	1.4.c
//P	Patch card.	2.2.d
//BX	Write a bootable record X = B if data in bootable format = O if data in object format	2.2.b (3.2.a)
//MXY	Specifies output media for master X = A for ARMM C for commercial tape Y = commercial tape 0-3 = blank for ARMM	2.2.a
//OX	Write a callable object record X = B if data in bootable format = O if data in object format	2.2.b
//CO	Convert compiler object to bootable cards.	3.2.b

//DXYZ	Process special PLANIT data
	X = I for input processing
	= O for output processing
	= E for end of input data
	Y = 0-3 for commercial tape 0-3 input
	= blank for cards if //DI, or MLU if //DO
	Z = 0-3 for commercial tape 0-3 output
	= blank for cards if //DO, or MLU if //DI
		4.2

//DNXX	Density to be used when writing commercial tape. XX = 08 for 800 BPI = 16 for 1600 BPI	4.2.1
//AMXN	Copy original library from commercial tape X to commercial tape N and perform update per card reader input during copy process.	5.2.a

(Note 1) CONTROLLING CHARACTERS	MEANING	REFERENCE MANUAL SECTION
//ARXX	Replace original program XX with new program in card reader.	5.2.b
//AAXX	Add new program in card reader before original program XX.	5.2.b
//ADXX	Delete original program XX.	5.2.b
//AE	Marks end of library update card reader input. Rest of library tape is copied.	5.2.c
//LSX	List summary of library tape mounted on commercial tape unit X.	6.2
//PPX	Punch PLANIT translation tape X = input tape number, 0-3.	7.2

//PCXY	Convert PLANIT translation tape from EBCDIC to ASCII and write a labeled tape for input to the L3050 librarian; list input and punch 'CARDS FILE.'
	X = input tape number, 0-3.
	Y = output tape number, 0-3.
		8.2

//CVXY	Convert data from one character set to another. X = Blank for card input = 0-3 for input tape number. Y = Blank for card output = 0-3 for output tape number.	9.2

NOTE 1: The control characters for PUP control cards must start in column 1. Except for the //P card, there must be at least 2 blanks following the last control character, at which point the balance of the card may be used for comments.

APPENDIX B

SAMPLE SUMMARIZE OUTPUT

```
 ****  // CARD LISTING   ****   S-14-D PLANIT OBJECT LIBRARY   REEL 0007

      229   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//80*     POS,0014              POSPOOL-9 NO ID      03-24-75
1 POS      0000 218E 028C          POS         POOL-9 032475       -4     034
//P
//R                                                                                        WARNING  GOES TO CARD READER
       164   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*     RAMCHECK,0010         POSPOOL-8 ID=10      12-04-74
1 RAMCHECK 0000 0506 0072        RAMCHECK     POOL-8 120474       -4     033
//P
0002 0000,0010 ID  RAMCHECK
//R                                                                                        WARNING  GOES TO CARD READER
        50   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*     MIOP,0020             POSPOOL-9 ID=20      03-24-75
1 MIOP     0000 1AA2 0588          MIOP        POOL-9 3-24-75      -4     034
//P
0020 0000,0020 MIOP ID
1400 0000,0000      MIOP PATCH R'3'      3-24-75
//R                                                                                        WARNING  GOES TO CARD READER
       220   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*     TMIOP,0008            PCSPOOL-9 ID=21      03-10-75
1 TMIOP    0000 063E 0360          TMIOP       POOL-9 031075       -4     033
//P
0002 0000,0021      TMIOP ID
08C2 0000,0000     TMIOP PATCH R'3'      3-10-75
//R                                                                                        WARNING  GOES TO CARD READER
        94   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*     PLANIT     03-12-75  (PLANIT3) ID=30
1 PLANIT   0000 3150 0032          PLANIT      PLNPOO 031275       -4     034
//P
0002 0000,0030 PLANIT ID
//R                                                                                        WARNING  GOES TO CARD READER
       406   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*     PLAN1      2-28-75   (PLANIT3) ID=41
1 PLAN1    0000 2670 006E          PLAN1       PLNPOO 022875       -4     034
//P
0002 0000,0041  PLAN1 ID
//R                                                                                        WARNING  GOES TO CARD READER
       315   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*     PLAN2      2-21-75   (PLANIT3) ID=42
1 PLAN2    0000 1562 0044          PLAN2       PLNPOO 022175       -4     034
//P
0002 0000,0042  PLAN2 ID
//R                                                                                        WARNING  GOES TO CARD READER
```

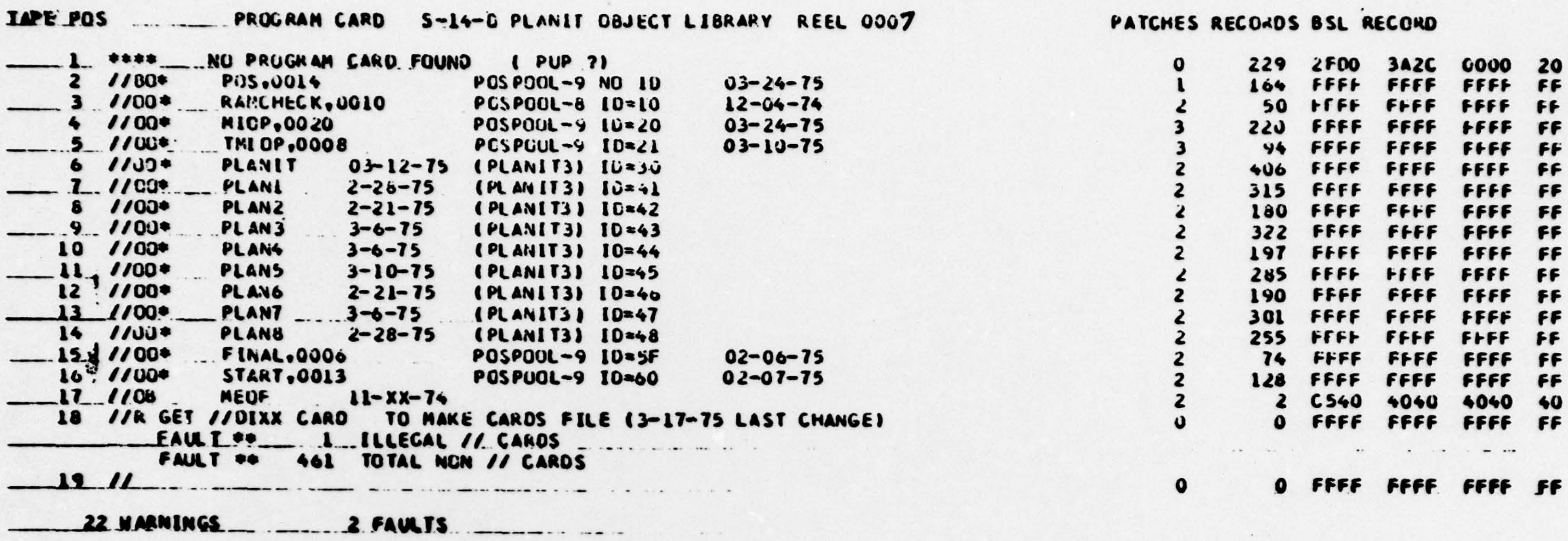
Table B-I. Sample Summarize Output (Sheet 1 of 4)

```
**** // CARD LISTING    ****    S-14-D PLANIT OBJECT LIBRARY  REEL 0007

      180   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*    PLAN3      3-6-75     (PLANIT3) ID=43
1 PLAN3    0000 26C2 002A        PLAN3          PLNPOO 030675        -4      034
//P
0002 0000.0043  PLAN3 ID
//R                                                                                         WARNING  GOES TO CARD READER
      322   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*    PLAN4      3-6-75     (PLANIT3) ID=44
1 PLAN4    0000 178A 0014        PLAN4          PLNPOO 030675        -4      034
//P
0002 0000.0044  PLAN4 ID
//R                                                                                         WARNING  GOES TO CARD READER
      197   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*    PLAN5      3-10-75    (PLANIT3) ID=45
1 PLAN5    0000 220E 0032        PLAN5          PLNPOO 031075        -4      034
//P
0002 0000.0045  PLAN5 ID
//R                                                                                         WARNING  GOES TO CARD READER
      285   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*    PLAN6      2-21-75    (PLANIT3) ID=46
1 PLAN6    0000 16F0 0032        PLAN6          PLNPOO 022175        -4      034
//P
0002 0000.0046  PLAN6 ID
//R                                                                                         WARNING  GOES TO CARD READER
      190   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*    PLAN7      3-6-75     (PLANIT3) ID=47
1 PLAN7    0000 2486 0032        PLAN7          PLNPOO 030675        -4      034
//P
0002 0000.0047  PLAN7 ID
//R                                                                                         WARNING  GOES TO CARD READER
      301   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*    PLAN8      2-28-75    (PLANIT3) ID=48
1 PLAN8    0000 1EB2 0020        PLAN8          PLNPOO 022875        -4      034
//P
0002 0000.0048  PLAN8 ID
//R                                                                                         WARNING  GOES TO CARD READER
      255   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//00*    FINAL,0006            POSPOOL-9 ID=5F      02-06-75
1 FINAL    0000 08CC 023A        FINAL          POOL-9 020675        -4      033
//P
0002 0000.005F FINAL ID
//R                                                                                         WARNING  GOES TO CARD READER
       74   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)
```

Table B-I. Sample Summarize Output (Sheet 2 of 4)

```
  **** // CARD LISTING    ****   S-14-0 PLANIT OBJECT LIBRARY  REEL 0007

//00*    START,0013          POSPOOL-9 ID=60     02-07-75
1 START   0000 OF8C 0280       START       POOL-9 020775          -4    033
//P
0002 0000,0060 START ID
//R                                                                                WARNING  GOES TO CARD READER
     128   PROGRAM RECORDS FOLLOWED BY       1   BLANK CARD(S)

//08     MEOF      11-XX-74
//P
0000 FFFF,FFFF,FFFF,FFFF        MACCAP EOF RECORD
        2   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//R GET //DIXX CARD   TO MAKE CARDS FILE (3-17-75 LAST CHANGE)                      WARNING  GOES TO CARD READER
C-0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ+-*/().%=,:;'¬ @|<?$                          WARNING 1ST NON // CARD BEFORE BLK OR PRG CARD
            ABCDEBCHAJCLMNABCRCDABWCAC                                     04000200 WARNING 2ND NON // CARD BEFORE BLK OR PRG CARD
1.$ '                                                                      04000300 WARNING 3RD NON // CARD BEFORE BLK OR PRG CARD
2.$ =                                                                      04000400 WARNING 4TH NON // CARD BEFORE BLK OR PRG CARD
** WARNING    457 NON // CARDS FOUND BEFORE NEXT CARD OR BLANK.  (ONLY 1ST FOUR ARE PRINTED, SEE ABOVE WARNINGS)
//0E                                                                                WARNING  ILLEGAL // CARD
        0   PROGRAM RECORDS FOLLOWED BY      1   BLANK CARD(S)

//
```

Table B-I. Sample Summarize Output (Sheet 3 of 4)



```
TAPE POS           PROGRAM CARD   S-14-D PLANIT OBJECT LIBRARY  REEL 0007                PATCHES RECORDS BSL RECORD

    1  ****    NO PROGRAM CARD FOUND     ( PUP ?)                                         0     229  2F00  3A2C  0000  20
    2  //00*      POS,0014                  POSPOOL-9 NO ID     03-24-75                  1     164  FFFF  FFFF  FFFF  FF
    3  //00*      RAMCHECK,0010             POSPOOL-8 ID=10     12-04-74                  2      50  FFFF  FFFF  FFFF  FF
    4  //00*      MIOP,0020                 POSPOOL-9 ID=20     03-24-75                  3     220  FFFF  FFFF  FFFF  FF
    5  //00*      TMIOP,0008                POSPOOL-9 ID=21     03-10-75                  3      94  FFFF  FFFF  FFFF  FF
    6  //00*      PLANIT      03-12-75      (PLANIT3) ID=30                               2     406  FFFF  FFFF  FFFF  FF
    7  //00*      PLAN1       2-26-75       (PLANIT3) ID=41                               2     315  FFFF  FFFF  FFFF  FF
    8  //00*      PLAN2       2-21-75       (PLANIT3) ID=42                               2     180  FFFF  FFFF  FFFF  FF
    9  //00*      PLAN3       3-6-75        (PLANIT3) ID=43                               2     322  FFFF  FFFF  FFFF  FF
   10  //00*      PLAN4       3-6-75        (PLANIT3) ID=44                               2     197  FFFF  FFFF  FFFF  FF
   11  //00*      PLAN5       3-10-75       (PLANIT3) ID=45                               2     285  FFFF  FFFF  FFFF  FF
   12  //00*      PLAN6       2-21-75       (PLANIT3) ID=46                               2     190  FFFF  FFFF  FFFF  FF
   13  //00*      PLAN7       3-6-75        (PLANIT3) ID=47                               2     301  FFFF  FFFF  FFFF  FF
   14  //00*      PLAN8       2-28-75       (PLANIT3) ID=48                               2     255  FFFF  FFFF  FFFF  FF
   15  //00*      FINAL,0006                POSPOOL-9 ID=5F     02-06-75                  2      74  FFFF  FFFF  FFFF  FF
   16  //00*      START,0013                POSPOOL-9 ID=60     02-07-75                  2     128  FFFF  FFFF  FFFF  FF
   17  //08       MEOF        11-XX-74                                                    2       2  C540  4040  4040  40
   18  //R GET //DIXX CARD   TO MAKE CARDS FILE (3-17-75 LAST CHANGE)                     0       0  FFFF  FFFF  FFFF  FF
            FAULT **     1  ILLEGAL // CARDS
            FAULT **   461  TOTAL NON // CARDS
   19  //                                                                                 0       0  FFFF  FFFF  FFFF  FF

       22 WARNINGS          2 FAULTS
```

Table B-I. Sample Summarize Output (Sheet 4 of 4)

APPENDIX C

COPYING TAPES

The LSS (L-3050 Support Software) tape contains the routines necessary to copy the various tapes which may be used or created by the PLANIT system. The following steps are necessary to initialize the copy routines:

a. Set the DATA EXCHANGE CHANNEL SELECT switches on IOU as follows:

   A = 1

   B = 2

   C = 3

b. Press INSTRUCTION STOP switch on CTS, should be illuminated.

c. Mount the LSS tape on commercial tape drive 0-3 (choice is at operator discretion).

d. Set PEBU CHANNEL SELECTOR switch to 1.

e. Set PEBU BSL SELECTOR switch to position for unit containing LSS tape (TU0 - TU1).

f. Press MASTER RESET switch on CTS.

g. <u>TOS^2 SSS only</u>. Press RESET switch on CTS.

h. Press BSL switch on PEBU. A good bootstrap load will be indicated by a DIG display of 777201, if not, check and repeat steps a through g. If condition persists, call maintenance personnel.

i. Change PEBU CHANNEL SELECTOR switch to 7.

j. Press INSTRUCTION STOP switch on CTS, light should extinguish.

k. Press START on CTS. The following messages will be output to typewriter:

```
LSS IS IN
AVAILABLE MEMORY =64K
ENTER DATE
```

l. Enter the date through the typewriter in the following format: dd/mm/yy. The following message will be output to the typewriter: ENTER TIME.

m. Enter the time through the typewriter in the following format: hhmm. The following message will be output to the typewriter: GO.

When the 'GO' message is output the system is ready to copy tapes. To copy any tape the same variable format is used no matter what type of tape used.

The copy format is: iCjqn for commercial tape copy and iKj for commercial to MLU copy where i, j, q and n are used as follows:

i = Input device (see list below)
j = Output device (see list below)
q = Blank, do not rewind or unload
  = R rewind when done
  = U rewind/unload when done
n = Number of files to copy (blank will copy one file)

Device identification is as follows:

TAPE 0	=	0
TAPE 1	=	1
TAPE 2	=	2
TAPE 3	=	3
ARMM1	=	8
ARMM2	=	9

If tape copy is to be performed using the ARMM(s), these devices must be physically cabled to DATA EXCHANGE CHANNEL A (IOX 1). Note: TOS^2 SSS only - Potter tape units must be off (or off-line) when ARMM (RMMU/MLU) is in use.

If an 800 BPI commerical tape is to be written TRANSFER SW2 on the CTS must be on before the tape copy is started.

At the conclusion of a copy operation the message "OK" will be output to the typewriter and the next copy may be performed.

APPENDIX D

SAMPLE JOBSTREAM CARDS

This appendix is included to illustrate several sample jobstream card decks to aid in understanding the various detailed sections of this manual. Specifically, the job decks delivered with the contractual program tapes are illustrated and described.

D.1 PLANIT Load Tape Generation

The PLANIT load tape generation deck illustrates the application of the information described in Sections 1, 2, 3, and 4 of this manual. It should be noted that a number of the "jobstream cards" have been included on the object library tape (see Appendix B and Table D-I) and therefore do not appear in the actual card deck. This accounts for the warning and fault messages in the Summarize Output shown in Appendix B. Table D-I illustrates the entire jobstream for creating a PLANIT load tape from the PLANIT Object Library tape.

The jobstream shown assumes that the PLANIT Object Library tape has been mounted on Commerical Tape Unit 2 (TU2) and a write enabled scratch tape (which will become a PLANIT load tape) has been mounted on Commercial Tape Unit 0 (TUO). Several jobstream options are also described. The generation deck described in Table D-I is loaded in the card reader. The hardware must be set up (PEBU BSL SELECTOR set to TU2), the PUP program loaded and the job initiated as described in Section 2.3.

TABLE D-I. PLANIT LOAD TAPE GENERATION DECK (Sheet 1 of 3)

JOB CARD CARDS NOTED ( ) ARE ON OBJECT TAPE	MEANING	REFERENCE MANUAL SECTION
//DN08	Load tape to be written at 800 BPI. [Note: Must be on a commercial tape drive with 800 BPI capability. Delete this card if 1600 BPI is desired (or change to / / DN16). Delete this card if load tape is to be output on an MLU.]	4.2.1
//MC0	Write load tape on tape unit 0. (Note: Delete this card if load tape is to be output on an MLU.)	2.2.a
//T2 POS	Go to tape unit 2 to start POS record processing.	1.4.a
(//B0)	Write bootable record from POS object module.	2.2.b 3.2.a
(// P)	POS patch cards to follow.	2.2.d
(// R)	Go to card reader for patch cards.	1.4.b
patch card(s)	POS patch cards as required (see Note 1).	2.2.d
//T2 RAMCHECK	Go to tape unit 2 to complete POS record and start RAMCHECK.	1.4.a
(blank card)	End of POS data.	4.2
(//00)	Write object record from RAMCHECK object module.	2.2.b
(//P)	RAMCHECK patch cards to follow.	2.2.d
(//R)	Go to card reader for patch cards.	1.4.b
patch card(s)	RAMCHECK patch cards as required (see Note 1).	2.2.d

TABLE D-I. PLANIT LOAD TAPE GENERATION DECK (Sheet 2 of 3)

JOB CARD CARDS NOTED ( ) ARE ON OBJECT TAPE	MEANING	REFERENCE MANUAL SECTION
//T2 MIOP	Go to tape unit 2 to complete RAMCHECK record and start MIOP.	1.4.a
	Etc. for TMIOP, PLANIT (Main), PLAN 1 thru Plan 8, FINAL and START program modules	
(//OO)	Write object record from START object module	2.2.b
(//P)	START patch cards to follow.	2.2.d
(//R)	Go to card reader for patch cards.	1.4.b
patch card(s)	START patch cards as required (see Note 1).	2.2.d
//T2	Go to tape unit 2 to complete START and write end of file (EOF) record.	1.4.a
(//OB)	Write EOF object record from EOF bootable tape record.	2.2.b
(//P)	EOF patch cards to follow, if applicable.	2.2.d
(//R)	Go to card reader for patch cards and to get //DIXX card to make "cards file."	1.4.b
patch card(s)	EOF patch cards, if applicable, (see Note 1).	2.2.d

TABLE D-I. PLANIT LOAD TAPE GENERATION DECK (Sheet 3 of 3)

JOB CARD CARDS NOTED ( ) ARE ON OBJECT TAPE	MEANING	REFERENCE MANUAL SECTION
//DI20	Write "cards file" on load tape (TUO) from "cards file" record on object tape (TU2). [Note: Change this card to //DI2 if the load tape is to be output on an MLU.] [Note: Change this card to //DI□2 or //DI (for MLU), add the "cards file" card deck and a //DE card if the "cards file" is to be read from cards instead of from the object library.]	4.2 (Blank)
(//DE)	End of "cards file" input data.	4.2
//	End of jobstream	1.4.c

Note 1: If no patches are required, there will be no cards in this portion of the jobstream deck.

D.2 Object Library "Cards File" Update

The object library "cards file" update deck illustrates the application of the information described in Sections 1, 4, 5, and 6 of this manual. Table D-II illustrates the entire jobstream for creating an updated PLANIT library tape with a new or revised "cards file."

The jobstream shown assumes that the original PLANIT Object Library tape has been mounted on Commercial Tape Unit 1 (TU1) and a write enabled scratch tape (which will become the new updated PLANIT object library tape) has been mounted on Commercial Tape Unit 2 (TU2). The update generation deck described in Table D-II is loaded in the card reader. The hardware must be set up (PEBU BSL SELECTOR set to TU1), the PUP program loaded and the job initiated as described in Section 5.4.

TABLE D-II. OBJECT LIBRARY "CARDS FILE" UPDATE GENERATION DECK

JOB CARD	MEANING	REFERENCE MANUAL SECTION
//AM12	Copy library tape onto tape unit 2 while updating per following job cards. Original library tape on tape unit 1.	5.2.a
//AR17	Replace program #17 (cards file) on original library tape with the cards to follow.	5.2.b
//R GET etc.	First cards to be stored on new library tape to allow load tape processing as described in D.1.	1.4.b
C-0123 etc. thru $$$$ etc.	PLANIT "cards file."	
//DE blank card	Last two cards to be stored on new library tape to allow load tape processing as described in D.1.	4.2
//AE	End of library update card input. Copy rest of library tape after program #17.	5.2.c
//LS2	List summary of new library tape on tape unit 2.	6.2
//	End of jobstream	1.4.c

APPENDIX E

GLOSSARY OF TERMS

A

ACC - (TACFIRE) Artillery Control Console

ACCCE/ACCCED - (ACC) Compose Edit (lower) Display

ACCRD - (ACC) Receive (upper) Display

ACCSA - (ACC) Switch Assembly

ARMM - Auxiliary Removable Media Memory Unit (includes MLU)

B

BOT - Beginning of Tape

BSL - Bootstrap Load

C

CE/CED - Compose Edit Display on the ACC or OCC (Same as ACCCED
(C/E) or OCCCED)

CPU - AN/GYK-12 Computer Central Processing Unit

CTS - Computer Test Set

D

DDT - Digital Data Terminal

DE - Display Editor

DIG - (CPU) Diagnose Status Lights

DPM - Digital Plotter Map (not used with PLANIT)

E

ELP - Electronic Line Printer

EOT - End of Tape

ETD - Electronic Tactical Display (not used with PLANIT)

F

FI - Fault Isolation program(s), part of AN/GYK-12 system software (not a part of PLANIT system)

FINAL - AN/GYK-12 PLANIT System Final (termination of PLANIT System operations) program module

FSU - (MIOD) Format Storage Unit (not used with PLANIT)

H

HSP - High Speed Printer (commercial peripheral printer in SSS and PSSB)

I

IOU - AN/GYK-12 Computer Input/Output Unit

K

KB - Alphanumeric Keyboard

L

LSS - L-3050 Support Software (General Utility Programs)

M

MADCAP - Maintenance and Diagnostic Control and Activation Program (MADCAP Operating System used as basic building block for POS)

MCMU - Mass Core Memory Unit (131 k words, 32 bits plus parity per word)

MEOF - MADCAP End of File

MIOD - (TOS^2) Message Input/Output Device

MIOP - Machine Input/Output Program

MLU - Memory Load Unit (part of ARMM, includes TTC)

O

OCC - (TOS^2) Operator Control Console

OCCCE/OCCCED - (OCC) Compose Edit (lower) Display

OCCRD - (OCC) Receive (upper) Display

OCCSA - (OCC) Switch Assembly

P

PCG - Power Converter Group

PEBU - Peripheral Equipment Buffer Unit

PLAN 1 - PLANIT Overlay 1 Program Module

PLAN 2 - PLANIT Overlay 2 Program Module

PLAN 3 - PLANIT Overlay 3 Program Module

PLAN 4 - PLANIT Overlay 4 Program Module

PLAN 5 - PLANIT Overlay 5 Program Module

PLAN 6 - PLANIT Overlay 6 Program Module

PLAN 7 - PLANIT Overlay 7 Program Module

PLAN 8 - PLANIT Overlay 8 Program Module

P
(Continued)

PLANIT - Programming Language for Interactive Teaching
Note: This is also the name given to the PLANIT MAIN program module.

POS - PLANIT Operating System

PSSB - (TACFIRE) Programming Support System B

PUP - PLANIT Utility Program

R

RAM - Random Access Memory (drum)

RAMCHECK - RAM track check program

RAMFI - RAM Fault Isolation program, part of AN/GYK-12 system software (not a part of PLANIT system)

RD - Read-only Display on the ACC or OCC (same as ACCRD or OCCRD)

RMMU - Removable Media Memory Unit (Same as ARMM)

S

SA - (ACC/OCC) Switch Assembly

SSS - (TOS^2) Software Support System

START - AN/GYK-12 PLANIT System Start program module

T

TACFIRE - Tactical Fire Direction System (U. S. Army Artillery)

TACPOL - Tactical Procedure Oriented Language (programming language for the AN/GYK-12 computer)

TMIOP - Terminal MIOP

TOS^2 - Tactical Operating System Operable Segment

TTC - (ARMM/MLU) Tape Transport Cartridge

V

VFMED - (TACFIRE) Variable Format Message Entry Device